Apple II



Manuale del sistema Operativo D.O.S 3.3

Disk Operating System

AVVISO

La Apple Computer Inc. si riserva il diritto di apportare miglioramenti al prodotto descritto in questo manuale in qualsiasi momento e senza preavviso.

ESCLUSIONE DI TUTTE LE GARANZIE E RESPONSABILITÀ

La Apple Computer Inc. non formula alcuna garanzia nè espressa nè implicita rispetto a questo manuale nè rispetto al software in esso descritto, alla sua qualità, alle sue prestazioni, commerciabilità o idoneità per qualsiasi particolare scopo. Il software della Apple Computer Inc. è venduto o ceduto in licenza "così com'è ,,.
L'intero rischio per quanto riguarda la sua qualità e le sue prestazioni è a carico dell'acquirente. Se i programmi risultano difettosi dopo il loro acquisto, l'acquirente (e non la Apple Computer Inc., o il suo distributore o il suo rivenditore) si assume l'intero costo di tutte le riparazioni necessarie, manutenzione o correzione e di qualsiasi danno conseguente o incidente. In nessun caso la Apple Computer Inc. potrà essere ritenuta responsabile dei danni diretti, indiretti, incidenti o conseguenti dovuti a qualsiasi difetto nel software, anche se la Apple Computer Inc. è stata informata della possibilità di tali danni. Alcuni paesi non consentono l'esclusione o la limitazione delle garanzie implicite o delle responsabilità per danni incidenti o conseguenti, per cui la suddetta limitazione o esclusione potrebbe non applicarsi al caso specifico.

Apple Computer International
7, rue de Chartres
92200 Neuilly-sur-Seine
France

Numero di codice APPLE E2LT036 (030-T115-00)

Apple II

Manuale del sistema Operativo D.O.S 3.3

Disk Operating System

Indice

Prefazione

CAPITOLO 1
Installazione e manutenzione

CAPITOLO 2
Per cominciare

CAPITOLO 3
Uso delle opzioni

CAPITOLO 4
Operare in modo sicuro

CAPITOLO 5
Altre informazioni di "preparazione"

CAPITOLO 6
Uso dei files sequenziali

CAPITOLO 7
Apple "Automatico"

CAPITOLO 8
Uso dei files ad accesso casuale

CAPITOLO 9
Uso dei files in linguaggio macchina

CAPITOLO 10
Input, Output e Concatenamento

APPENDICE A
Tipi di files usati con i comandi DOS

APPENDICE B
Messaggi DOS

APPENDICE C
Formato delle informazioni su mini floppy

APPENDICE D
Uso della memoria

APPENDICE E
Punti di entrata DOS e schemi

APPENDICE F
Sommario dei comandi DOS

APPENDICE G
Sommario delle procedure DOS

APPENDICE H
Aggiornamento del DOS per ottenere 16 sectori

APPENDICE I
Uso del mini floppy BASICS

APPENDICE J
Uso del programma FID

APPENDICE K
Uso del programma MUFFIN

Indici

Prefazione

Questo manuale ha due funzioni principali. La prima è di insegnare ad usare il DOS (Disk Operating System): i capitoli del manuale si servono di esempi per accompagnare le spiegazioni sul modo in cui funzionano i vari comandi DOS.

La seconda funzione del manuale è di servire come guida di riferimento al DOS.

Le Appendici, la Tabella di riferimento rapido, gli indici sono stati studiati tenendo presente questa funzione.

Per usare un Apple Disk II, occorre un computer Apple II con almeno 16 K di memoria – anche se si consigliano 32 K, dato che il sistema a 16 K lascia poco spazio di memoria per caricare i programmi. Infatti, i programmi MUFFIN e FID residenti nel System Master Diskette, richiedono almeno 32 K.

Apple Disk II è un'unità mini floppy, che consente di caricare e richiamare informazioni molto più rapidamente e comodamente che non con il nastro. Le informazioni sono caricate e richiamate da un "mini floppy", un piccolo disco di plastica particolarmente trattato (diametro circa 12,7 cm), permanentemente sigillato in un involucro quadrato pure di plastica.

Uno dei principali vantaggi nell'uso di Disk II è che le informazioni sono caricate e richiamate col nome con cui vengono archiviate. Un programma che cataloga numeri di telefono protebbe essere memorizzato con una istruzione tipo
SAVE NUMERI TELEFONICI
e richiamato con un comando altrettanto semplice. Il nome NUMERI TELEFONCI con cui il programma è archiviato è il **nome file.**

I programmi che controllano automaticamente i files, memorizzano e richiamano le informazioni e svolgono inoltre numerosi altri compiti di manutenzione, sono chiamati **Disk Operating System** (sistema operativo del disco), solitamente abbreviato in "DOS", che alcuni pronunciano "DOSS", e altri pronunciano "di o esse". Per imparare l'uso di DOS e del disco occorre conoscere alcuni speciali comandi descritti in questo manuale. Questi comandi possono essere usati come estensioni dei programmi in linguaggio macchina o dell'Integer BASIC o Applesoft.

In alcuni punti si troverà il simbolo.



che precede un paragrafo. Questo simbolo indica una caratteristica insolita alla quale bisogna fare attenzione.

Il simbolo



precede paragrafi che descrivono situazioni dalle quali il BASIC può non essere in grado di uscire. In questo caso si perde il programma e probabilmente occorre far ripartire il DOS e anche il BASIC.

*** **NOTA** ***

Questo Manuale fa riferimento alla versione DOS che opera su 16 settori, cioè al DOS 3.3 e versioni successive. Dato che le versioni precedenti operano su 13 settori, sarà necessario sostituire due proms del controller, come specificato in Appendice H, per effettuare l'aggiornamento. Potrete usare poi il programma MUFFIN, per convertire i vostri precedenti programmi memorizzati a 13 settori, copiandoli su nuovi mini floppy a 16 Settori. Per lanciare (in boot) mini floppy da 13 Settori su un sistema da 16, usate il dischetto BASICS che troverete nella confezione dell'unità Disk II o nel Disk II Update Kit.

In alcune circostanze, un mini floppy contenente programmi convertiti con MUFFIN può funzionare non correttamente. Ciò accade quando il programma convertito utilizza specifiche locazioni fisiche sul mini floppy per riferimento o proprio utilizzo. È un caso abbastanza insolito ma può capitare, e in tal caso sebbene MUFFIN effettui una corretta conversione il programma non funzionerà.

CAPITOLO 1
Installazione e manutenzione

Apertura dell'imballo

Il sistema Disk II si compone di diverse parti:

1) L'unità disco (la scatola principale)
2) Una scheda (la scheda dell'unità di controllo) che si inserisce nell'Apple II
3) Un cavo a nastro piatto già fissato all'unità disco, per il collegamento di questa alla scheda dell'unità di controllo
4) Un mini floppy "BASICS"
6) Questo manuale

Se è stata acquistata soltanto l'unità disco (ad esempio come seconda unità disco per la scheda dell'unità di controllo), il sistema non comprenderà tutte le suddette voci.

Conservare il materiale di imballo nel caso si desideri trasportare il disco – o nell'improbabile caso in cui si debba ritornarlo al rivenditore o alla fabbrica per la riparazione. Inviare la cartolina di garanzia – ciò non solo pone in atto la garanzia, ma fa inserire il nome del proprietario nell'elenco, per la distribuzione degli aggiornamento.

*** **NOTA SPECIALE** ***

Prima di collegare o scollegare
QUALSIASI COSA
su Disk II o Apple II,
TOGLIERE L'ALIMENTAZIONE
Questa è una **necessità imprescindibile**

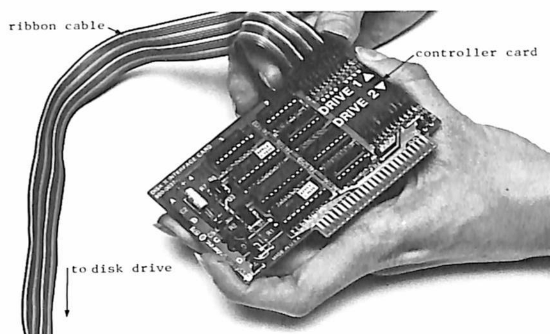
Collegamento del cavo

Nell'uso, l'unità disco verrà collegata ad un'unità di controllo su scheda, mediante il cavo piatto tipo nastro. Un'estremità di questo cavo è già fissata all'unità disco. Se questa è la prima unità disco che viene usata il connettore all'estremità del cavo a nastro uscente dall'unità deve essere fissato alla serie **superiore** di spine nella scheda dell'unità di controllo. Questa serie di spine dell'unità di controllo contrassegnata "DRIVE 1".

*** **ATTENZIONE** ***

Se il cavo uscente dall'unità disco in direzione della scheda dell'unità di controllo non è inserito correttamente in quest'ultima, può provocare seri danni all'unità disco ed alla sua parte elettronica. Per assicurare il corretto montaggio, inserire il cavo nella scheda dell'unità di controllo **prima** di installarla nel computer. Seguono ora due consigli di installazione. Innanzitutto non schiacciare o ripiegare il cavo tra il connettore e la scheda dell'unità di controllo. Se il cavo è inserito dovrebbe uscire dal suo connettore sul lato di quest'ultimo lontano dalla scheda dell'unità di controllo, come indicato nella fotografia.

Secondo, assicurarsi che **tutte** le spine del connettore della scheda dell'unità di controllo entrino nei corrispondenti fori nel connettore del cavo a nastro. Effettuando il collegamento prima di installare la scheda, si può chiaramente **vedere** che tutte le spine entrino correttamente nei rispettivi fori.



Collegamento del cavo all'unità di controllo
Didascalie della figura :
1 Cavo a nastro
2 Scheda dell'unità di controllo
3 All'unità disco

Se si installa una **seconda** unità disco, occorre collegare il cavo a nastro dalla seconda unità alla serie inferiore di spine dell'unità di controllo. Questa serie di spine è contrassegnata "DRIVE 2". Collegare questo connettore con la stessa cura usata per il primo.

Installazione dell'unità di controllo

Per installare la scheda dell'unità di controllo Disk II, che è già stata collegata all'unità disco attraverso il cavo a nastro, basta inserirla in una presa all'interno di Apple II, come segue :

1. **Spegnere l'interruttore** posto nella parte posteriore di Apple II. Ciò è importante per impedire danni al computer. Con la tensione inserita, la rimozione o l'inserimento di qualsiasi scheda può provocare danni permanenti sia alla scheda che a Apple II.

2. Rimuovere il coperchio da Apple II. Ciò si effettua sollevando il coperchio in corrispondenza del bordo posteriore (il bordo più lontano dalla tastiera), fino a che i due dispositivi di fissaggio agli angoli non si staccano. A questo punto, far scorrere il coperchio all'indietro, fino a che non si libera.

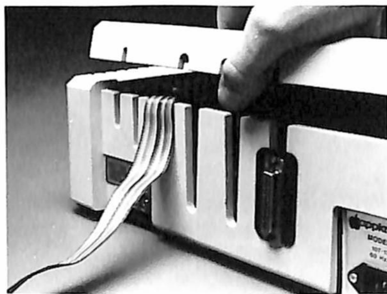
3. All'interno di Apple II, in corrispondenza della parte posteriore della scheda, c'è una fila di 8 lunghe prese strette denominate "slots". La più a sinistra guardando il computer dal lato tastiera) è lo slot Ø e quello più a destra è lo slot 7. Individuare lo slot 6. La scheda dell'unità di controllo può essere disposta in qualsiasi presa (slot) salvo la Ø ossia quella più a sinistra. In ogni caso, la posizione standard di Apple per la scheda dell'unità di controllo del disco è lo slot 6 e la maggior parte del software Apple (e questo manuale) è scritto tenendo presente questa posizione.

4. ASSICURARSI CHE L'APPARECCHIO SIA SPENTO PRIMA DI INSERIRE O RIMUOVERE QUALSIASI SCHEDA DAL COMPUTER. Inserire la parte a pettine dell'unità di controllo nello slot 6. La parte a pettine entra nella presa con un certo attrito e quindi si insedia saldamente. Dato che i identi del pettine fanno contatto elettrico è bene evitare di toccarli con le dita. Prima dell'installazione, può essere utile sfregarli con alcool per pulire le impronte digitali sulla scheda.



Inserimento della scheda
dell'unità di controllo

5. Regolare il cavo, in modo che si disponga in piano e passi sopra una delle zone tra le aperture verticali nella parte posteriore di Apple II, come indicato nel disegno. Quando il coperchio è installato, blocca il cavo e agisce anche da serrafilo.



Posizionamento del cavo

6. Rimontare il coperchio di Apple II, iniziare facendo scorrere il bordo anteriore nel coperchio in posizione e premere sui due angoli posteriori fino a che il coperchio non si blocca.

7. L'unità di controllo Disk II è installata e Apple II può essere acceso. Disporre l'unità disco in un posto comodo, solitamente di fianco o sopra ad Apple II.

Installazione di più unità disco

Ciascuna scheda unità di controllo può essere usata con due unità disco, una collegata alla serie superiore di contatti contrassegnati "DRIVE 1" e la seconda alla serie inferiore contrassegnata da "DRIVE 2". La prima unità disco deve essere collegata ai contatti DRIVE 1 e la seconda ai contatti DRIVE 2 sulla scheda nello slot 6. La terza e la quarta unità disco devono essere collegate rispettivamente ai contatti DRIVE 1 e DRIVE 2 su una scheda nello slot 5, la quinta e la sesta unità del disco rispettivamente ai contatti DRIVE 1 e DRIVE 2 su una scheda nello slot 4, e cosi via.

Se si dispone di più unità disco, vale la pena di contrassegnare la parte anteriore di ciascuna unità con il numero di unità e di slot, dato che i programmi faranno riferimento ai dischi mediante questi numeri.

Cura di disk II e dei mini floppy

L'unità Disk II, a differenza di Apple II, è un dispositivo meccanico, con motori e parti mobili e pertanto, è alquanto più delicata di un computer. Il trattamento violento, come il lasciar cadere l'apparecchio o appoggiarvi degli oggetti pesanti può provocarne il danneggiamento. L'unità non deve essere disposta di fianco o sopra un televisore, dato che i forti campi magnetici emessi dagli apparecchi televisivi possono provocare danni alle proprietà magnetiche del l'unità. In caso di dubbio, disporre le unità disco 60-70 cm dal televisore.

Ciascun mini floppy è un piccolo disco di plastica (di diametro circa 12,7 cm) trattato in modo particolare da poter accogliere le informazioni e cancellarvele dalla sua superficie. Lo strato superficiale è simile al rivestimento magnetico di un nastro di registrazione. Il mini floppy è permanentemente sigillato in un coperchio di plastica nero quadrato, che lo protegge, lo tiene pulito e gli consente di girare liberamente. Questa confezione non deve mai essere aperta.

Il termine "floppy" deriva dal fatto che il mini floppy è flessibile. I dispositivi di memoria di informazione dei computer più vecchi, che lavoravano su analoghi principi usavano dischi rigidi. Anche se il mini floppy (e il suo coperchio di plastica) sono alquanto flessibili, in effetti la piegatura può danneggiarli. La confezione del mini floppy contiene sia agenti lubrificanti che autopulenti per estendere la durata del disco - trattare quindi queste confezioni con tutto rispetto.

Non permettere che nulla tocchi la superficie marrone o grigia del mini floppy, e manipolarlo afferrandolo soltanto per la busta di plastica nera. Quando il mini floppy non vieve usato, lasciarlo nella busta di carta con la quale viene fornito. Queste buste sono trattate per ridurre al minimo l'accumulo di elettricità statica che attira la polvere. Vale la pena di conservare i mini floppy verticalmente, quando non vengono usati.

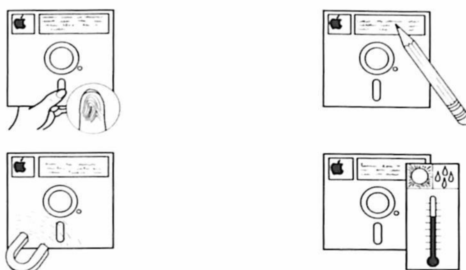
I mini floppy contengono un'enorme quantità di informazioni : un singolo mini floppy può contenere fino a 1.146.ØØØ bits di informazioni. Un singolo bit di informazione, pertanto, occupa una piccolissima porzione sul mini floppy. Un graffio invisibile sulla superficie, o addirittura un'impronta digitale può provocare errori. Non disporre i mini floppy su superfici sporche o unte; non lasciare che raccolgano polvere.

Per scrivere sull'etichetta di un mini floppy, usare un pennarello con punta di feltro e non premere molto. È bene non scrivere sull'etichetta fissata, ma sull'etichetta staccata e fissarla successivamente al mini floppy.

Tenere i mini floppy lontani dai campi magnetici. Ciò significa tenerli lontani dai motori eletrici e magneti e non disporli nella parte superiore dei dispositivi elettronici, tipo apparecchi televisivi. Essi possono per contro essere temporaneamente appoggiati su Apple II o su Disk II.

I mini floppy sono sensibili alle temperature estreme. Tenerli al riparo dal sole e lontani da altre sorgenti di calore che potrebbero provocarne l'ondulazione e/o fare perdere i dati. Nei giorni caldi, l'interno delle auto (o dei cruscotti delle auto) può essere dannoso. I mini floppy funzionano soddisfacentemente fino a 50°C, ossia ad una temperatura non molto elevata. La prima traccia di danno dovuta al calore è un'ondulazione o deformazione del coperchio di plastica nero.

Con ragionevole cura, un mini floppy ha una durata media di 40 ore continue di lavoro - un periodo estremamente lungo, quando si consideri che occorrono pochi secondi per caricare (LOAD) la maggiore parte dei programmi. Con un po'di attenzione, un mini floppy può non dare assolutamente alcun problema.



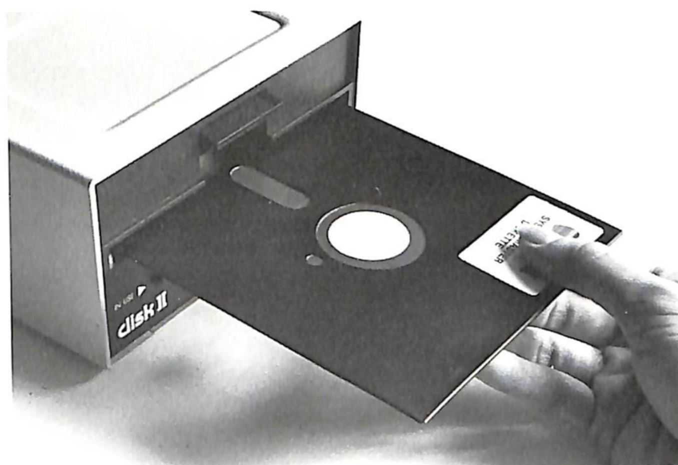
Le cose da non fare

Inserimento e rimozione dei mini floppy

L'uso di un'unità disco è molto più rapido e più facile che non l'uso di un registratore a cassetta, ma è necessaria una certa cura per proteggere i mini floppy. L'unità disco stessa deve essere manipolata con una certa cura. Lo sportello del disco si apre afferrandolo sul bordo inferiore e tirandolo verso l'esterno. Il mini floppy viene quindi inserito nella cavità **con l'etichetta rivolta verso l'alto** come indicato nella fotografia. Il bordo del mini floppy con la fessura ovale nella protezione di plastica quadrata deve inserirsi nell'unità per primo. Il bordo del mini floppy con l'etichetta deve entrare per ultimo.

Una buona regola elementare

Tenere il mini floppy con il pollice destro sull'etichetta : ciò aiuta notevolmente il corretto orientamento quando si inserisce nell'unità.



Inserimento di un mini floppy

Spingere il mini floppy **delicatamente** fino a che non è completamente inserito nell'apparecchio. Non piegarlo. Se lo si spinge rudemente, il mini floppy può venir permanentemente danneggiato. Chiudere lo sportello dell'unità disco spingendolo di nuovo verso il basso. I due denti metallici (che si possono vedere all'interno della cava quando viene chiuso la sportello) devono appena sfiorare il mini floppy quando lo sportello si chiude.

Il mini floppy viene rimosso aprendo lo sportello dell'unità ed estraendolo delicatamente dal unità stessa. L'atto di apertura dello sportello solleva la testina dal disco. Se si pensa di lasciare in un'unità per parecchie ore un mini floppy non utilizzato, è bene aprire lo sportello in modo che la testina non appoggi.



NON RIMUOVERE MAI un mini floppy mentre è accesa la spia "IN USE". Ciò può dannegiare permanentemente il mini floppy e quasi sicuramente distruggere le informazioni su di esso memorizzate. In tal caso, il mini floppy può solitamente essere riutilizzato, ma non è possibile recuperare le informazioni perse.

CAPITOLO 2

Per cominciare

Qualche notizia in generale

L'apprendimento dell'uso del disco el del suo sistema operativo consiste nell'imparare alcune istruzioni speciali, parecchie delle quali sono semplici estensioni delle familiari istruzioni BASIC.Questo manuale presume che si abbia già familiarità con Apple II e che si trovi facile scrivere semplici programmi in BASIC.

Per imparare ad usare Apple II e Integer BASIC, consultare il Manuale di programmazione integer BASIC Apple II. Per imparare ad usare BASIC Applesoft, consultare il Manuale di riferimento del BASIC Applesoft. Il manuale Applesoft presume che l'utente abbia già familiarità con Apple II e la semplice programmazione BASIC. Se questa familiarità non esiste con l'uno o con l'altro dei manuali, per prima cosa occorre conoscere di più Apple II prima di passare ad imparare il DOS.

Nel corso del manuale, figurano i listati di programma che illustrano come usare DOS. La maggior parte di questi programmi sono in Applesoft, alcuni sono in Integer BASIC. Talvolta, vengono citate le modifiche necessarie per convertire un programma Applesoft in Integer BASIC, altre volte no. Consultare l'appendice A nel manuale Applesoft per i particolari sulle differenze tra i programmi scritti in Integer BASIC e Applesoft BASIC.

Una piccola esperienza di prima mano vale a volte più della lettura. Una volta collegata l'unità disco e acceso il computer, seguire ciascuna di queste descrizioni tentando effettivamente di svolgere le procedure su Apple II.

Disporre il mini floppy System Master nell'unità disco. Questa parte del manuale tratta soltanto con un'unità disco e presume che siano seguite le condizioni standard, inserendo l'unità di controllo nello slot 6.



Quantunque i comandi DOS **sembrino** uguali ai comandi BASIC, non seguono tuttavia le stesse regole. Ad esempio, non è possibile inserire più comandi DOS su una riga, separati da virgole. In questo caso, risulterebbe un messaggio di errore di sintassi (SYNTAX ERROR).

Tasti special

Talvolta questo manuale usa le parentesi graffe {e} per racchiudere i nomi di tasti speciali che si suppone si debbano premere sulla tastiera Apple II.

{**RETURN**} significa che occorre premere il tasto contrassegnato **"RETURN".** Premere il tasto **RETURN** dopo ciascuna istruzione.

{**RESET**} significa premere il tasto **"RESET".** La pressione del tasto **RESET** interrompe l'esecuzione del programma.

{**ESC**} significa premere il tasto contrassegnato **"ESC"**, che significa originariamente "uscita" ma che ora ha altri significati.

{**CTRL**} è un poco diverso. Esso significa che bisogna premere il tasto contrassegnato **"CTRL"** (che sta per controllo) e continuare a tenerlo abbassato mentre si batte un altro tasto. Ad esempio {**CTRL**} **C** significa battere il tasto "C" mentre si tiene abbassato il tasto **CTRL**. Talvolta, l'uso di questo tasto è indicato in un altro modo: **CTRL-C** e {**CTRL**}**C** significano la stessa cosa.

*** **NOTA** ***
I caratteri battuti mentre si tiene abbassato il tasto **CTRL**
non compaiono sullo schermo.

Caricamento del DOS (Boot the DOS)

Il processo di aggiungere i comandi DOS al BASIC nell'Apple II è detto lancio del disco. Il disco può essere lanciato da Integer BASIC, da Applesoft o da Monitor. Ci sono vari modi in cui è possibile lanciare il DOS. Con Integer BASIC o Applesoft, possono essere usati i comandi PR#e IN#(vedere il manuale Applesoft).

Con Monitor, possono essere usati i "comandi di controllo" con il tasto **CTRL.** Una volta lanciato, il DOS è sempre la stesso e non ha importanza come lo si è lanciato.

Negli esempi che seguono, la lettera s sta per il numero dello slot di Apple II in cui l'unità di controllo disco è sistemata. La posizione standard per l'unità di controllo è la slot 6 (vedere capitolo 1, Installazione dell'unità di controllo). Dopo uno dei seguenti comandi, occorre sempre premere il tasto **RETURN.**

Con Integer BASIC (il cui carattere di richiesta è>) è possibile usare uno di questi comandi per eseguire il lancio del disco:

	PR#s	Esempio: PR#6
oppure	IN#s	Esempio: IN#6

Con Applesoft (il cui carattere di richiesta è]) è possibile usare uno o l'altro di questi comandi per eseguire il lancio del disco:

	PR#s	Esempio: PR#6
oppure	IN#s	Esempio: IN#6

Con Monitor (il cui carattere di richiesta è *) è possibile usare uno di questi comandi per eseguire il lancio del disco:

	Cs∅∅G	Esempio: C6∅∅G
oppure	s {**CTRL**} **P**	Esempio: 6 {**CTRL**} **P**

Nota : Il DOS viene comunque caricato automaticamente all'accensione, nei computers Europlus.

Nel resto di questo manuale, quando si deve far ripartire il DOS in questo modo si dirà semplicemente "lanciare il DOS" oppure "lanciare il disco". Entrambe le espressioni (molto diffuse tra gli utenti di computer) significano la stessa cosa. "Lancio" è un'abbreviazione dell'espressione "lancio iniziale del programma".

Proveremo ora ad eseguire il lancio iniziale del DOS dal mini floppy System Master. Iniziare mettendo Apple II in BASIC - ossia Integer BASIC o Applesoft. Assicurarsi che il mini floppy sia correttamente inserito. Battere ora:

```
PR#6
```

e premere il tasto **RETURN** come al solito. Da questo momento in poi, si presumerà che dopo ogni istruzione venga premuto il tasto **RETURN.**

Una volta premuto il tasto **RETURN**, si accende la spia rossa "IN USE" e il disco produce dei rumori tipo frullio e colpi secchi (non occorre allarmarsi - non è ancora pronto a decollare) e in meno di 1Ø secondi compare un messaggio. Il messaggio dovrebbe essere simile a quello che segue:

```
DOS VERSION 3.3          Ø4/15/8Ø
APPLE II STANDARD        SYSTEM MASTER
```

Se si tenta ora di usare il BASIC, si trova che la maggior parte dei comandi funziona ancora normalmente, e a parte la comparsa improvvisa dei messaggi, Apple II sembra invariato. Invece, è successo che alcuni comandi sono stati inseriti e quelli vecchi hanno assunto nuove capacità. Sono state apportate due modifiche che non sono ovvie, comunque:

1) Il puntatore HIMEM alla posizione di memoria più elevata che è possibile usare è stato ripristinato per accogliere il programma DOS.

2) Apple II può aver perso alcune delle sue capacità per grafici ad alta risolvenza, in relazione alla quantità di memoria nel computer. Per i particolari, vedere Appendice D sull'uso della memoria DOS.

** Le versioni del DOS che usano 13 Settori non possono essere lanciati con un sistema da 16. Per utilizzare i mini floppy da 13 Settori potete creare un nuovo floppy da 16, utilizzando il programma MUFFIN. Inoltre potete utilizzare i floppy da 13 Settori lanciando il mini floppy BASICS; leggere per maggiori informazioni l'appendice I.

Se il caricamento non funziona

Se non è possibile eseguire il lancio del mini floppy System Master, occorre rileggere il manuale accuratamente - il manuale risolve infatti il 90% di tutti i problemi.

Non è molto probabile, ma se l'apparecchio è stato spedito in un carro armato Sherman o in qualcosa del genere, i connettori all'interno dell'unità disco possono essersi leggermente allentati. Se si ha timore di frugare all'interno dell'apparecchio, affidarlo al concessionario che sarà lieto di farlo.

Se invece si ha la passione di mettere le dita nelle cose complicate, è possibile spegnere il computer e scollegare l'unità disco dall'unità di controllo. L'allentamento delle 4 viti sul fondo dell'unità disco consente al meccanismo di scivolare, fuori dal contenitore. Serrare i connettori premendoli dolcemente sulle schede del circuito stampato. Rimontare l'unità che probabilmente sarà ora in grado di funzionare. Se questo intervento di emergenza non funziona, rivolgersi al concessionario. Non eseguire alcuna regolazione.

Inizializzazione di nuovi mini floppy

Il System Master che viene fornito insieme a questo manuale è un mini floppy speciale: esso contiene programmi che consentono di copiare un intero mini floppy e aggiornare qualsiasi mini floppy che contenga una precedente versione del DOS e altro. Sul mini floppy figurano i programmi che dimostrano le varie capacità del DOS, programmi che sono discussi nel manuale.

Estrarre il mini floppy System Master dall'unità e sostituirlo con un altro mini floppy vuoto fornito insieme all'apparecchio. Tentare ora un esperimento.
Mettere in funzione il BASIC quindi battere

PR#6

e vedere cosa succede. Si accende la spia rossa IN USE e l'unità disco emette i soliti rumori, quindi inizia a girare dolcemente e silenziosamente e non si ferma. Per fermarlo, occorre premere il tasto **RESET** (normalmente questa è una cattiva idea, ma queste circostanze non sono normali). È invece una buona idea aprire lo sportello dell'unità disco prima di premere il tasto **RESET,** dato che la manovra solleva la testina dell'unità disco dalla superficie del mini floppy.

È successo questo: Apple II è andato invano alla ricerca di informazioni su un mini floppy vuoto (su un disco di questo genere è possibile cercare all'infinito...). Quando viene fabbricato, un nuovo mini floppy non contiene alcuna informazione, come del resto un nastro vergine comprato per un registratore. Per poter funzionare nel computer, nel mini floppy devono esserci alcune particolari informazioni : il mini floppy deve essere cioè **inizializzato.**

Se si è seguita la procedura dell'esempio, il mini floppy vuoto è nell'unità ed è stato premuto il tasto **RESET.** Estrarre ora il mini floppy vuoto, sostituirlo con il System Master e chiudere lo sportello dell'unità disco. Mettere il computer in BASIC e battere ancora

PR# 6

Si dovrebbe ottenere di nuovo il messaggio che era comparso al momento del lancio iniziale. Una volta ancora al BASIC sono stati aggiunti i comandi DOS.

Può essere usato il comando INIT per inizializzare un mini floppy "secondario". I mini floppy secondari dipendono dalla dimensione della memoria: la dimensione di un sistema sul quale viene inizializzato un mini floppy determina la dimensione del sistema che può usare il disco. Se viene creato un mini floppy secondario su un sistema da 32 K questo può essere quindi usato soltanto su sistemi da 32 K o di maggiori dimensioni. Ma su sistemi più grandi, vanno usati soltanto 32 K di memoria. Dopo l'inizializzazione di un mini floppy secondario è possibile usare il programma MASTER CREATE (vedere Capitolo 5) per trasformare il mini floppy secondario in un "master" il cui DOS è autorilocante e che permette che la memoria venga usata più efficacemente.

Il comando INIT richiede l'uso di un programma BASIC detto programma "greeting" (saluti) in quanto saluta l'utente: ogniqualvolta si esegue il lancio del mini floppy questo programma viene eseguito automaticamente. Il programma greeting è comunemente denominato "HELLO" ma lo si può anche chiamare "BUONGIORNO" o "GUTEN TAG" o come si preferisce. Semplifica la vita usare un nome standard per i programmi greeting quando si inizializzano i mini floppy.

Ecco una guida passo-passo per inizializzare un mini floppy secondario. Si supponga che il DOS sia già stato lanciato come descritto sopra.

1) Rimuovere il System Master dall'unità disco e sostituirlo con un mini floppy vuoto.

2) Battere NEW, quindi scrivere un programma greeting. Ecco un semplice esempio di un programma greeting:

```
5 REM GREETING–1 PROGRAM
1Ø PRINT "SLAVE DISKETTE CREATED ON 48K SYSTEM"
2Ø PRINT "BY AMY DOAKS ON 8 AUGU ST 1982"
3Ø END
```

Occorre indicare il nome, la dimensione del sistema, la data corrente e altre informazioni per aiutare a determinare rapidamente e facilmente la storia del mini floppy e la condizione secondario/principale.
È possibile eseguire **(RUN)** il programma per vedere se in effetti fa ciò che da esso ci si aspetta.

3) Se il programma è soddisfacente, battere questa istruzione:
INIT HELLO
Quando si preme il tasto **RETURN** il mini floppy gira per circa 2 minuti, facendo i soliti piccoli rumori. Quando l'inizializzazione è terminata, compare l'appropriato carattere di richiesta (ad esempio] per Appelsoft).

4) Quando il disco si ferma e la spia IN USE si spegne, rimuovere il mini floppy e contrassegnarlo. L'etichetta deve essere del tipo
32K SLAVE DISKETTE CREATED 8 AUGUST 198Ø

in modo che con una sola occhiata si possa sapere che non è vuoto.

Mettere da parte il minifloppy System Master fornito insieme al computer Apple e disporlo in un posto dove non sia danneggiato dal calore, da sollecitazioni fisiche (Bambini, cani) o oggetti magnetici. E dove non possa andar perso. Va trattato con particolare attenzione, dato che contiene molti programmi utili.

Una volta che un mini floppy è stato inizializzato, sarà indicato come secondario. Per contrassegnare il mini floppy secondario, occorre estrarlo dall'unità. Reinserirlo e tentare di eseguire il lancio iniziale: dovrebbe comparire il messaggio contenuto nelle istruzioni PRINT. Se si sono eseguite le istruzioni precedentemente indicate, sullo schermo dovrebbe comparire:

```
SLAVE DISKETTE CREATED ON 48K SYSTEM
BY AMY DOAKS ON 8 AUGUST 1982
```

Dato che il mini floppy – una volta vuoto – può ora essere lanciato, si sa che è stato inizializzato correttamente. Da questo punto in avanti è possibile usare il mini floppy secondario di recente inizializzato per fare esperimenti. Non è però possibile eseguire delle procedure che verranno dimostrate più avanti sul System Master in quanto il mini floppy "è protetto contro la scrittura", come discusso nel Capitolo 4.

Se sono stati acquistati altri mini floppy vuoti, sarebbe una buona idea inizializzarne ora qualcuno.

Caricamento e memorizzazione con DOS

Lanciare il sistema con il mini floppy inizializzato. Scrivere

NEW

per assicurarsi che non ci siano programmi in memoria. Ciò cancella il programma che è stato caricato (LOAD) ed eseguito **(RUN)** quando è stato lanciato il DOS.

Scrivere ora questo semplice programma:

```
 5 REM COUNT PROGRAM
10 FOR I = 1 TO 10
20 PRINT I
30 NEXT I
40 END
```

Eseguirlo **(RUN)** una anche due volte per assicurarsi che funzioni come previsto. In Applesoft quando il programma viene eseguito, si osserva quanto segue:

```
 1   2   3
 4   5   6
 7   8   9
10
```

A scopo di riferimento, chiamare questo programma ONE TO TEN, dato che esso conta da 1 a 10. Per caricare questo programma sul mini floppy scrivere l'istruzione

```
SAVE ONE TO TEN
```

Quando si termina, premendo il tasto RETURN, il disco emette i soliti rumori per alcuni secondi e il programma viene caricato.

Se è stato battuto SAVE senza alcun nome, il programma viene memorizzato sulla cassetta come al solito (supponendo che il registratore a nastro sia stato fatto funzionare come descritto nel Manuale di programmazione BASIC).
Per dimostrare che il programma è stato memorizzato (SAVE) sul mini floppy eseguire quanto segue: Innanzitutto battere LIST quindi **RUN** per vedere se il programma è ancora in memoria e se funziona ancora correttamente. Ciò dimostra che l'uso del DOS per memorizzare (SAVE) un programma su un mini floppy non influisce in alcun modo sul programma stesso.
Ora battere NEW quindi LIST. Non rimarrà alcun programma – i programmi sono scomparsi battendo NEW. Per assicurarsi che il programma è scomparso, spegnere il computer. È possibile estrarre il mini floppy e reinserirlo (delicatamente). Riaccendere di nuovo il computer. Battere NEW (che cancella il programma HELLO) quindi LIST. Non succede nulla? Giusto.

Ora scrivere

LOAD ONE TO TEN

e il disco gira per circa due secondi. Listare (LIST) il programma, lo si troverà in perfetta condizione. Ciò è tutto quanto riguarda la memorizzazione (SAVE) e il caricamento (LOAD) dei programmi da disco: è esattamente come per la cassetta, salvo che viene usato un nome file e l'operazione è più veloce.

Catalogo

Il programma ONE TO TEN è stato caricato sul mini floppy. Per vedere quali programmi sono caricati su un dato disco, battere il comando

CATALOG

per far comparire un elenco di tutti i programmi del mini floppy. Finora il catalogo del mini floppy dovrebbe apparire come il seguente se i programmi sono stati scritti in Integer BASIC:

```
I ØØ2 HELLO
I ØØ2 ONE TO TEN
```

La lettera "I" nella colonna di sinistra significa che i programmi sono in Integer BASIC. Prima dei nomi dei programmi Applesoft compare invece una "A". Oltre ai files di programma in BASIC ci sono anche altri tipi di files che possono essere memorizzati, che verranno spiegati nei Capitoli da 6 a 9. I numeri dopo la lettera del tipo di file rappresentano la lunghezza del programma memorizzato. In questo caso, per caricare il programma sono stati necessari ØØ2 "settori" di mini floppy. Ciascun settore può accogliere fino a 256 bytes di informazioni. Il file più breve possibile, un file di testo vuoto (vedere Capitolo 6) richiede ØØ1 settori per registrare talune informazioni di "manutenzione". In tutto, un mini floppy può accogliere 4Ø3 settori di programmi e altri files. Infine, ciascuna entrata nel catalogo contiene il nome del programma. Vedere l'appendice C per i particolari sul modo in cui le informazioni vengono caricate sui mini floppy.



Quando un file supera 255 settori, la lunghezza segnalata da CATALOG per quel file riparte da ØØØ.



Guardando CATALOG non c'è modo di sapere qual'è il programma greeting. Cosi è di aiuto se al programma greeting si dà sempre la stesso nome. Per cambiare il nome al programma greeting, vedere la discussione sul programma UPDATE, nel Capitolo 5.

Talvolta, ci sono più programmi su un mini floppy che compaiono contemporaneamente sullo schermo. CATALOG fa sì che vengano elencati i primi 18 programmi. Quando si è pronti ad esaminare gli altri programmi, premere qualsiasi tasto salvo **RESET,** il tasto **CTRL** o i tasti **SHIFT.**

Cosa c'è in un nome?

I nomi file devono avere una lunghezza compresa fra 1 e 3Ø caratteri; DOS tronca a 3Ø caratteri i nomi file più lunghi. Un nome file deve iniziare con una lettera. Nel nome può comparire qualsiasi carattere stampabile, salvo la virgola (,).

Ecco alcuni nomi file leciti:

SOMNAMBULISTICS
ONE TO TEN
HIRES 34
THE QUALITY OF MERCY: UNSTRAINED

Ecco alcuni nomi che **non** funzionó	(e il motivo per cui non funzionano)
1 TO 1Ø	(inizia con una cifra)
HI THERE,BABE	(contiene una virgola)
INEPT EXCESS VERBIAGE DISQUALIFIES NAMES	(verrà ridotto a 30 caratteri)

Quantunque il nome dell'ultimo file venga ridotto a 3Ø caratteri quando presentato da CATALOG, è possibile (e si può fare) battere l'intero nome nel caricamento (LOAD) o nell'esecuzione **(RUN)** e tutto funziona correttamente.

Ogni riga nel catalogo rappresenta un "file". Il programma BASIC che è stato caricato è un esempio di un file. Le regole qui indicate per i nomi file si applicano anche ai nomi dei programmi.



Se in un nome viene battuto accidentalmente un carattere di controllo (o anche di proposito) quel carattere non compare sullo schermo quando si richiama il catalogo. Ad esempio, se si batte {**CTRL**}**T** invece della semplice "T" nel nome "AGATHA", il listato di catalogo comparirebbe come:

AGAHA

Per contro, se si tenda di caricare quel file battendo

LOAD AGAHA

il computer risponde

FILE NOT FOUND

Quantunque il nome che è stato battulo sembrasse identico al nome nel catalogo. Quindi bisogna fare attenzione: non bisogna inserire inavvertitamente caratteri di controllo nei nomi file.

Il capitolo Nomi File dell'Appendice F contiene suggerimenti sul modo in cui trovare quali caratteri di controllo sono incorporati nei nomi file.

Cambio nome dei files

Per un motivo o un altro, si desidera di tanto cambiare il nome di un file. Si supponga di essersi stancati di battere il nome file ONE TO TEN e di decidere di chiamare quel file

RENAME ONE TO TEN, COUNT

dopo un attimo, comparirà il carattere di richiesta BASIC.
Battere ora

CATALOG

per verificare che tutto si sia svolto come desiderato.



Il comando RENAME non controlla se il nuovo nome che si desidera usare già esiste o meno, per cui è possibile ridenominare (RENAME) fino a che tutti i files su un mini floppy hanno lo stesso nome... una situazione indesiderabile e di massima confusione che sarebbe meglio evitare.

Cancellazione dei files

È facile rimuovere i files dal mini floppy. Battere di nuovo

CATALOG

per vedere i due files presenti sul mini floppy. Battere ora

LOAD COUNT

(supponendo di aver cambiato il nome file come sopra descritto) per avere quel programma in memoria. Cancellare questo programma dal mini floppy mediante l'istruzione

DELETE COUNT

e verificare che la concellazione sia andata a buon fine battendo

CATALOG

Viene lasciato soltanto il programma greeting – probabilmente chiamato HELLO –. Dato che il programma COUNT è in memoria (perchè lo si è caricato – LOAD), è possibile riportarlo sul mini floppy con il comando ormai noto

SAVE COUNT

Dare ancora uno sguardo al catalogo per assicurarsi che il programma sia di nuovo sul mini floppy. Se si tenta di cancellare (DELETE) un file che non si trova sul mini floppy, si ottiene il messagio

FILE NOT FOUND

Recupero da interruzioni accidentali

Supponiamo che in Integer BASIC o in Applesoft non ci sia il DOS. (Se Applesoft è in firmware si suppone che l'interruttore sulla scheda sia predisposto per Applesoft). Anche il DOS prevede delle procedure di recupero che solitamente conservano programmi e dati.

Se è già stato effettuato il lancio iniziale del DOS, e quindi si preme **RESET,** si ottiene la richiesta del Monitor (*). Per ritornare al DOS e al BASIC battere

3DØG

Ricordare che tra le lettere D e G c'è uno zero e non la lettera O.

Se il recupero al DOS non funziona, e il programma è listabile (LIST) non tutto è perso: Caricare il programma su nastro (sperando che ci si sia ricordati di tenere disponibile l'unità nastro per tale emergenza), quindi, è possibile lanciare DOS, caricare il programma da nastro (LOAD) e memorizzarlo (SAVE) su un mini floppy.

Se si è battuto accidentalmente o intenzionalmente il tasto **RESET** mente la spia rossa "IN USE" del floppy è accesa, le informazioni sul mini floppy **possono** essere distrutte. I problemi si verificano con più facilità quando si stanno inserendo informazioni sul mini floppy, usando un comando SAVE, BSAVE o WRITE.

Se le informazioni vengono distrutte, probabilmente non è possibile recuperare i programmi dal mini floppy. Se null'altro funziona, è possibile reinizializzare il mini floppy e usarlo di nuovo, ma l'inizializzazione (INIT) distrugge **tutti** i files.

Se la spia IN USE rimane accesa per parecchi minuti ma non si ode il solito frullio del disco, il sistema può essere "bloccato". la pressione di **RESET** può essere il solo modo per spegnere la spia e far ripartire il sistema.

Un mini floppy può essere parzialmente rovinato, cosìcchè non se ne può effettuare il lancio, ma in tale circostanza è possibile lanciare talvolta **un altro** mini floppy, quindi caricare i programmi dal mini floppy parzialmente rovinato e memorizzarli su uno non danneggiato. oppure, si può usare il programma Fid per copiare files o programmi da salvare su un mini floppy affidabile.

CAPITOLO 3

Uso delle opzioni

Opzione slot, drive e volume

La maggior parte dei comandi DOS consente di specificare un certo numero di options, tipo ad esempio quale unità disco si usa, quale slot accoglie l'unità di controllo del disco per quel l'unità e un "numero di volume" per il floppy.

L'option dell'unità disco consente di operare con più di un'unità. Ciascuna unità di controllo ha la possibilità di pilotare una o due unità disco. Normalmente, le istruzioni si riferiscono all'unità 1. Questo è il valore **presunto:** se non si specifica l'unità floppy, viene comunque usata l'unità 1. Se si desidera specificare l'unità 2, occorre usare la notazione D2 separata dal nome file o da altre options del disco mediante una virgola. Ad esempio, per inizializzare un mini floppy nell'unità 2, è possibile usare l'istruzione

INIT HELLO, D2

Dopo aver specificato l'unità 2, **tutti gli altri ulteriori comandi** del disco si riferiranno all'unità 2, fino a che non viene di nuovo specificata l'unità 1. L'unità 2 è ora l'unità presunta. Dopo la suddetta inizializzazione, il comando

CATALOG

elenca i files caricati sul mini floppy nell'unità 2. Per specificare l'unità 1, usare la notazione D1 separata dal nome file mediante una virgola. Ad esempio

CATALOG, D1

indica i contenuti del mini floppy nell'unità 1 e cambia il numero presunto dell'unità riportandolo a 1.

Se vengono usate più di due unità disco, occorrono ulteriori unità di controllo. Queste sono disposte in slots diverse da quello della prima unità di controllo (che è abitualmente nello slot 6). È possibile specificare lo slot n (dove n è una cifra da 1 a 7), con la notazione Sn separata da una virgola dal nome file e da altre option del disco. Ad esempio, per inizializzare un mini floppy nell'unità 1 collegata ad un'unità di controllo nello slot 5 si deve usare l'istruzione

INIT HELLO, S5, D1

Il nome file viene per primo, mentre l'ordine delle options non è importante.

Il numero di slot presunto è quello usato al momento del lancio del DOS. Una volta specificato un diverso numero di slot, questo diventa il valore presunto fino a che non viene espilicitamente cambiato.



Se si usa un comando DOS con un parametro che indica uno slot che non contiene un'unità di controllo del disco, si ottiene un messaggio

I/O ERROR

e tutto sembra procedere bene. Ma il DOS pensa ora che il numero di slot presunto sia quello sbagliato e che il floppy che non è collegato a quello slot è ancora in funzione. Anche se il successivo comando DOS specifica lo slot **giusto**,

esso attende **per sempre** il floppy non esistente. Se non ci sono in memoria programmi che si vogliono conservare, basta rilanciare il DOS. Per recuperare il programma intatto, procedere come segue:

1. Ripristinare lo slot presunto battendo

   CATALOG, Ss

dove s è il numero corretto di slot.

2. Quando il sistema si blocca, premere il tasto **RESET.**

3. Scrivere

   CALL-151

4. Battere

   3DØG

e tutto dovrebbe riprendere a funzionare regolarmente.



Il DOS **deve** essere lanciato da un mini floppy nell'unità 1 e non nell'unità 2.

La option del numero di volume può essere usata per proteggere i mini floppy dalla cancellazione accidentale, mediante sovrascrittura. Ad esempio, si supponga di avere un sistema di inventario basato su mini floppy, dove le registrazioni di ciascun mese siano su un diverso mini floppy con un unico numero di volume. Quando ci si accinge ad impostare le informazioni per il mese di gennaio, occorre essere sicuri di specificare il corretto numero di volume, altrimenti le informazioni non vengono scritte sul mini floppy e si ottiene un messaggio

   VOLUME MISMATCH

Un "numero di volume"può essere assegnato a un mini floppy quando questo viene inizializzato (INIT) usando la notazione Vn separata da una virgola dal nome file o da altre options del disco. Ad esempio, per inizializzare un mini floppy che usa il nome "START UP" per il programma greeting (programma che viene eseguito ogniqualvolta il mini floppy viene lanciato) dove il mini floppy è nel Drive 2 di un'unità di controllo nello slot 5, l'assegnazione di un numero di volume di 128 comporta l'uso del comando.

   INIT START UP, D2, S5, V128



Il numero di volume di un mini floppy non può essere cambiato senza reinizializzare il mini floppy stesso.

Le options del numero di unità del numero di slot e del numero di volume possono comparire in qualsiasi ordine. Il suddetto comando è equivalente a

   INIT START UP, V128, S5, D2

e a

   INIT START UP, S5, V128, D2

e cosi via

Il numero di volume di un mini floppy deve essere un numero untero da 1 a 254. Se con INIT non è specificato alcun numero di volume, viene assegnato un numero di volume presunto 254.



Il comando

INIT HELLO, VØ

non dà alcun messaggio ma assegna al mini floppy il numero di volume presunto 254.

Tutti i comandi DOS possono specificare il numero di volume, se si desidera che il DOS controlli che il numero di volume sul mini floppy concordi con la option V. Se non si spécifica alcun numero di volume o se si specifica il volume zero oppure se si batte "V" senza numero, il DOS ignora il numero di volume del mini floppy. Se si specifica accidentalmente un numero di volume scorretto, il sistema lo respinge con il messaggio

VOLUME MISMATCH

Gli errori di abbinamento di volume non posso verificarsi quando si chiede di vedere il CATALOG. Il numero di volume di un mini floppy compare all'inizio dell'elenco CATALOG.

Le options vengono ulteriormente discusse quando viene presentato ciascun comando. Inoltre, le informazioni per ciascun comando sono riassunte nell'Appendice Sommario dei comandi e sulla Scheda di riferimento rapido che accompagna questo manuale. I seguenti paragrafi spiegano come interpretare questi concisi sommari.

Sintassi

Sintassi si riferisce alla struttura di un comando di computer, all'ordine e alla forma corretta delle varie parti del comando stesso. Per descrivere la sintassi di ciascun comando DOS viene usata una semplice notazione. Le voci tra parentesi quadre ([e]) sono facoltative; le parti facoltative di un comando DOS possono essere specificate in qualsiasi ordine. Le lettere maiuscole e le virgole devono sempre essere battute come indicato; le lettere minuscole stanno per le voci che occorre indicare. Nella specifica della sintassi per i comandi DOS

f   sta per nome file
d   sta per numero unità disco – 1 o 2
s   sta per numero slot – da 1 a 7
v   sta per numero volume – solitamente da 1 a 254

Un numero di volume di mini floppy non può essere Ø. La specifica di un numero di volume Ø in un comando di disco dice al DOS di ignorare il numero di volume.

Le ulteriori abbreviazioni usate in questo manuale sono riassunte all'inizio dell'Appendice del Sommario dei comandi.

Qualsiasi costante numerica (il numero dell'unità disco, il numero del volume, ecc.) in un comando DOS può essere espressa in notazione esadecimale facendo precedere le cifre esadecimali dal segno del dollaro. Se non si conosce la notazione esadecimale, ignorare l'istruzione precedente – non occore infatti conoscere la notazione esadecimale per comprendere questo manuale.

INIT

La sintassi per il comando INIT è

INIT f [,Vv] [,Ss] [,Dd]

dove le parentesi quadre indicano le options che possono o meno essere incluse. L'esempio

INIT HELLO, V17, D2

Può essere interpretato come segue.

Il nome del comando "INIT" è in maiuscolo e deve essere battuto esattamente come indicato. La "f" del nome file è sostituita dal nome file legittimo "HELLO". Successivamente è indicato il numero di volume optional: Prima viene una virgola, quindi, la lettera maiuscola "V". In questo esempio, la "v" per il numero del volume è stata arbitrariamente sostituita da 17. Le parentesi quadre introno a ",Ss" indicano che la spécifica del numero di slot è optional per il comando INIT: in questo esempio è stato omesso, per cui il DOS userà il numero di slot presunto. È prevista invece la option della unità disco: la virgola e la lettera maiusola "D" devono essere come indicato; in questo esempio la lettera "d" è sostituita da 2. Per i particolari sull'uso di INIT, vedere al Capitolo 2 "Inizializzazione di nuovi mini floppy".

LOAD, RUN e SAVE

Il caricamento (LOAD), l'esecuzione (**RUN**) e la memorizzazione (SAVE) di programmi sul disco sono simili alle corrispondenti operazioni per la cassetta (salvo che i programmi sono indicati per nome file) ma tutto si svolge ad una velocità 10 volte superiore e non occorre mai premere alcun bottone per riprodurre, registrare o riavvolgere. È tutto automatico. Ci sono molte altre capacità che caratterizzano un disco, come ad esempio il catalogo dei programmi e l'esecuzione automatica dei programmi senza intervento dell'utente. Anche la memorrizzazione dei dati (sui files di testo – vedere Capitolo 6) è molto facile.

È una buona idea basarsi sul sistema a nastro per scambiare i programmi e come memoria di riserva per programmi e dati importanti (quantunque l'esperienza dimostra che la memoria su disco per programmi e per dati sia ancora più affidabile che non la memoria su nastro).

Se c'è un programma in BASIC e si desidera chiamarlo HENRY, il comando

SAVE HENRY

lo memorizza su mini floppy. Se c'è più di un'unità, HENRY normalmente viene memorizzato sull'unità disco dalla quale è stato lanciato inizialmente il DOS (l'unità presunta, a meno che non si sia specificata una unità diversa dopo il lancio iniziale). È possibile specificare il numero dell'unità, il numero di volume e il numero di slot come con il comando INIT. Ad esempio, per memorizzare (SAVE) un file denominato AGATHA sull'unità 1 dell'unità di controllo nello slot 2, dove il numero di volume del mini floppy è 214, è possibile usare il comando

SAVE AGATHA, D1, S2, V214

Come già detto, le tre options possono essere messe in qualsiasi ordine. Se viene omessa la option del numero di volume, AGATHA viene memorizzata ugualmente, ma il DOS non controlla se il mini floppy ha il numero di volume 214.

I nomi dei programmi sono nomi file e devono seguirne le stesse regole. Essi possono avere una lunghezza massima di 30 caratteri e devono iniziare con una lettera. Possono comprendere qualsiasi carattere salvo le virgole o i caratteri di controllo. Ecco alcuni nomi validi per i files:

CHECKBOOK
THE QUALITY OF MERCY
HIRES 34
NOW: HEAR THIS!

Per caricare (LOAD) un programma denominato AGATHA, usare il comando

LOAD AGATHA

e il programma di quel nome, se nel catalogo ce n'è uno, viene caricato. Per provare se AGATHA, è caricato, eseguirlo.

Se si desidera che AGATHA venga eseguita (RUN) dopo che è stato caricato, è possibile naturalmente usare i comandi

LOAD AGATHA

quindi

RUN

ma c'è anche un modo per farlo in una sola fase:

RUN AGATHA

è un comando DOS che prima carica (LOAD) il file specificato quindi lo esegue (**RUN**)

Ecco la sintassi per i comandi SAVE, LOAD et **RUN**

SAVE f [,Ss] [,Dd] [,Vv]
LOAD f [,Ss] ,Dd] [,Vv]
RUN f [,Ss] [,Dd] [,Vv]

Seguono alcuni esempi:

SAVE OUR HAPPY HOME, D1, S7
LOAD UP
RUN AMOK, S7



Se, **quando si tenta di memorizzare (Save) un programma** si nota un messaggio SYNTAX ERROR, significa che è stato compiuto un errore di battuta o che il DOS non è stato correttamente lanciato. Innanzitutto occorre provare a ribattere il comando. Se il DOS era stato inizialmente lanciato, ed otterete ancora SYNTAX ERROR, provate a rilanciare il DOS. Se il DOS non era stato lanciato - NON RILANCIATELO. **Il lancio del DOS cancellerà qualsiasi programma in memoria.** Innanzitutto memorizzare il programma su nastro usando il solito comando per la cassetta.

SAVE

Ora rilanciate il DOS. Successivamente, usare il solito comando per la cassetta

LOAD

per riportare il programma nella memoria di APPLE II dal nastro.
Ora è possibile memorizzarlo su disco.
Se un mini floppy è difettoso (probabilmente qualcuno ha tentato di pinzarlo in un libro di appunti), oppure se il mini floppy non è stato inizializzato, oppure se non c'è mini floppy nell'unità, o se lo sportello è aperto, compare il messaggio

I/O ERROR

(I/O sta per input/output) quando si tenta di memorizzare (SAVE) o caricare (LOAD) servendosi del DOS. Controllare tutte le voci elencate e correggere il problema. Non occorre rilanciare di nuovo il DOS. Tentare, invece, di nuovo.

Se si usa il comando

LOAD HENRY

ed HENRY non è il nome di un programma sul mini floppy inserito nell'unità, si ottiene il messaggio

FILE NOT FOUND

Esaminare il catalogo del mini floppy per trovare l'esatto nome file del programma. Tutti i caratteri e gli spazi devono essere battuti esattamente come compaiono nel nome file indicato nel catalogo.

DELETE

Per eliminare qualsiasi file che non si desidera avere sul mini floppy, può essere usato il comando

DELETE

La sintassi è

DELETE f [,Ss] [,Ds] [,Vv]

Ad esempio il comando

DELETE EXCESS, V34, D2, S1

cancella un file denominato EXCESS da un mini floppy con il numero di volume 34, che è posto nell'unità 2, dell'unità di controllo nella slot 1. I settori su un mini floppy sono definiti "liberi" soltanto quando un file è CANCELLATO (DELETE).

Un esempio: lancio, SAVE (memorizzazione), RUN (esecuzione)

CATALOG (catalogo), DELETE (cancellazione)
Si supponga di operare in Integer BASIC e che il mini floppy System Master sia nell'unità disco. Qui c'è il dialogo che potrebbe comparire sullo schermo di Apple II. Le parti battute dall'utente sono sottolineate, quantunque esse non compaiano in quel modo sullo schermo.

```
>PR #6
```

(quanto sopra cancella lo schermo ed è possibile osservare quanto segue:)

```
DOS VERSION 3. 3                  04/15/80
APPLE II STANDARD                 SYSTEM MASTER

>CATALOG
DISK VOLUME 254
*A ØØ6 HELLO
*I Ø18 ANIMALS
*I ØØ3 APPLE PROMS
*I ØØ6 APPLESOFT
*I Ø26 APPLEVISION
*I Ø17 BIORHYTHM
*B Ø1Ø BOOT13
*A ØØ6 BRIAN'S THEME
*B ØØ3 CHAIN
*I ØØ9 COLOR DEMO
*A ØØ9 COLOR DEMOSOFT
*I ØØ9 COPY
*B ØØ3 COPY OBJØ
*H ØØ9 COPYA
*H Ø1Ø EXEC DEMO
*B Ø2Ø FID
*B Ø5Ø FPBASIC
*B Ø5Ø INTBASIC
*H Ø28 LITTLE BRICK OUT
*H ØØ3 MAKE TEXT
*B ØØ9 MASTER CREATE
*B Ø27 MUFFIN
*R Ø51 PHONE LIST
*A Ø1Ø RANDOM
*A Ø13 RENUMBER
*A Ø39 RENUMBER INSTRUCTIONS
*A ØØ3 RETRIEVE TEXT

>NEW
>1Ø print ''JABBERWOCK''
>2Ø END
>SAVE DEMO
WRITE PROTECTED
```

←[A questo punto si vuole inserire il mini floppy secondario precedentemente inizializzato, dato che non è protetto contro la scrittura.]

```
>CATALOG
DISK VOLUME 254
I ØØ2 HELLO
I ØØ2 COUNT
>SAVE DEMO
>CATALOG
DISK VOLUME 254
I ØØ2 HELLO
I ØØ2 COUNT
i ØØ2 DEMO
>NEW
>RUN
*** NO END ERR
>RUN DEMO
JABBERWOCK
>DELETE DEMO
>CATALOG
DISK VOLUME 254
I ØØ2 HELLO
I ØØ2 COUNT
```

Cambiamento di linguaggi: FP E INT

Si supponga di usare Integer BASIC e di voler scrivere un programma in Applesoft, oppure di usare il computer come calcolatore con numeri in virgola mobile (numeri con punti decimali). Per richiamare APPLESOFT senza rovinare il DOS, battere

FP

e in un attimo Applesoft è pronto a funzionare.
FP sta per "Floating Point" (virgola mobile), naturalmente. (Se per qualche motivo Applesoft non è disponibile – non è in firmware o nel mini floppy in uso – compare il messaggio

LANGUAGE NOT AVAILABLE

La sintassi per il comando è

FP [,Ss] [,Dd]

dove i parametri optional della slot e dell'unità disco consentono di specificare l'unità che contiene Applesoft su mini floppy.
Se si sta usando Applesoft DOS è possibile battere

INT

(per "Integer BASIC") per ritornare all'Integer BASIC con il DOS intatto. La sintassi per questo comando è semplicemente

INT

senza parametri. Si genera un messaggio
SYNTAX ERROR
SE SI TENTA DI USARE I PARAMETRI D o S con INT.



Se si batte

INT

mentre si è in Integer BASIC, si perde qualsiasi programma nella memoria; analogamente se si batte

FP

mentre si è in Applesoft, si perde qualsiasi programma in memoria.

Quando si passa da Integer BASIC ad Applesoft o viceversa, si perde qualsiasi programma che in quel momento si trova in memoria.
Oltre a spostarsi tra i BASIC di Apple, è possibile passare in Monitor ed usare i comandi DOS.
Per far ciò da Applesoft o da Integer BASIC battere

CALL –151

in modo da ottenere*, il carattere di richiesta del Monitor. Per ritornare a qualsiasi BASIC dal quale si era partiti con il programma e con il DOS intatto, battere

3DØG



Dal Monitor, si può anche battere

INT

per ritornare all'Integer BASIC, oppure

FP

per ritornare a Applesoft; in entrambi i casi, il DOS funziona ancora ma qualsiasi programma in memoria e scomparso.



Se si ottiene un messaggio

PROGRAM TOO LARGE

quando si tenta di eseguire un comando

FP

battere

    INT

per ripristinare il sistema. Quindi battere

    FP



Quantunque il mini floppy contenga il programma Integer BASIC denominato APPLE-SOFT, non battere

    RUN APPLESOFT

Se lo si fa, Applesoft sembra funzionare bene fino a che non si preme RESET, ad esempio, e si tenta di reinserire Applesoft.
In tal caso, dato che il DOS pensa di essere in Integer BASIC (in quanto Applesoft era un programma Integer BASIC) si va nei guai. Per spostare il programma Applesoft da un mini floppy ad un altro basta

    LOAD APPLESOFT

da un qualsiasi mini floppy sul quale si trovi, quindi disporre il mini floppy che si desidera contenga Applesoft nell'unità disco e battere

    SAVE APPLESOFT

Uso del DOS all'interno di un programma

Molto spesso è utile poter essere in grado di eseguire un comando DOS dall'interno di un programma BASIC. Ad esempio, si può desiderare che il programma greeting su un disco visualizzi il contenuto del disco eseguendo un comando CATALOG. Molti comandi DOS possono essere eseguiti dall'interno di un programma BASIC. Ciò avviene stampando (PRINT) una stringa che contiene un CTRL-D seguito dal comando.

Qui c'è un programma Applesoft che, se usato come programma greeting, fa si che le informazioni nelle istruzioni PRINT nelle righe 20 e 30 compaiano sullo schermo, seguite da un elenco di files nel CATALOG.

```
 5  REM GREETING PROGRAM
1Ø  D$ = CHR$ (4): REM CHR$(4)
    IS CTRL-D
2Ø  PRINT "SLAVE DISKETTE CREATED
    ON 32K SYSTEM"
3Ø  PRINT "BY AMY DOAKS ON 8 AUGU
    ST 198Ø"
4Ø  PRINT D$; "CATALOG"
5Ø  END
```

Il modo consigliato per far ciò in Applesoft è illustrato nel programma suddetto. Innanzitutto viene creata la stringa D$ composta soltanto da CTRL-D usando la funzione CHR$ nella prima riga del programma. Successivamente può essere usata come nella riga 4Ø

```
40 PRINT D$; "CATALOG"
```

Notare il punto e virgola dopo D$ e le virgolette intorno al comando DOS. Il punto e virgola è facoltativo nelle istruzioni PRINT di Applesoft, cosicchè se un programma contiene molti comandi DOS nelle istruzioni PRINT, si può risparmiare tempo e spazio di memoria semplicemente omettendole, ed usando la forma

```
40 print D$ "CATALOG"
```

In Applesoft, è possibile usare la funzione CHR$ per specificare CTRL-D

```
10 D$=CHR$(4): REM CTRL-D
```

Ma occorre ricordare che il codice ASCII per CTRL-D è 4, cosicchè può essere utile un REM (Nota). (La funzione CHR$ non è disponibile in Integer BASIC).

Sia in Integer BASIC che in Applesoft è possibile definire CTRL-D battendo i caratteri

```
D$="
```

quindi battendo la lettera D tenendo abbassato il tasto CTRL, e quindi battendo le virgolette.
Notare che CTRL-D non scrive sullo schermo. Il comando finale compare come

```
D$=""
```

Dato che i caratteri di controllo non vengono stampati, è spesso utile fare seguito con un REM (Nota) per ricordare cosa effettivamente c'è nella stringa. Ecco il suddetto programma scritto in Integer BASIC:

```
10  D$="": REM THERE IS AN INVISIBLE CTRL-D BETWEEN THE QUOTES
20  PRINT "SLAVE DISKETTE CREATED ON 32K SYSTEM"
30  PRINT "BY AMY DOAKS ON 8 AUGUST 1980"
40  PRINT D$; "CATALOG"
50  END
```

In un'istruzione PRINT può essere usato soltanto un comando DOS. L'istruzione PRINT deve iniziare con CTRL-D e terminare con il comando DOS.



L'uso della freccia a destra per copiare un'istruzione BASIC contenente un carattere di controllo invisibile cancella il carattere di controllo.



Nei comandi DOS eseguiti da un programma, il D$ deve essere preceduto da un RETURN oppure viene ignorato.
L'esecuzione di questo programma (RUN)

```
 5  REM TESTCATALOG PROGRAM
10  D$="": REM THERE IS AN INVISIBLE CTRL-D BETWEEN THE QUOTES
20  PRINT "TEST";
30  PRINT D$; "CATALOG"
40  END
```

provoca la comparsa di

TESTCATALOG

dato che il punto e virgola sopprime il **RETURN** al termine del comando PRINT nella riga 2Ø.
Per correggere questa situazione, e far si che venga eseguito anche il comando DOS CATALOG insieme al programma (**RUN**), basta cancellare il punto e virgola (;) dalla fine della riga 2Ø. Questi comandi DOS devono essere usati soltanto all'interno di programmi in un'istruzione PRINT che inizia con un CTRL-D:

OPEN
APPEND
READ
WRITE
POSITION

Questi comandi DOS possono essere usati nel modo ad esecuzione immediata, ed anche dall'interno di un programma usando un comando PRINT con CTRL-D:

CATALOG
SAVE
LOAD
RUN
DELETE
RENAME
LOCK E UNLOCK
MON E NOMON
BSAVE
BLOAD
BRUN
EXEC
CLOSE
CHAIN
PR #
IN #



Il comando DOS MAXFILES, può essere usato come descritto sopra in un programma Applesoft, ma deve essere usato in modo speciale in un programma Integer BASIC, come discusso nella parte che riguarda il comando EXEC, Capitolo 7.



Il comando DOS INIT deve essere usato soltano nel modo ad esecuzione immediata (possono verificarsi serie conseguenze se si ignora questa ammonizione).

CAPITOLO 4

Operare in modo sicuro

Sono già stati citati due modi per proteggere l'utente e/o mini floppy dai disastri. Il Capitolo 3 cita l'uso della option Volume per assicurarsi di inserire le informazioni sul supporto desiderato. L'uso dei caratteri di controllo nei nomi file può anche essere utile per proteggere l'utente (vedere Capitolo 2 "Cosa c'è in un nome?" ed inoltre l'Appendice F, "Nomi file"). Se ciò che compare nel CATALOG come

MY BANK ACCOUNT

in effetti ha le iniziali dell'utente disposte come carattere di controllo in alcuni punti nel nome, è improbabile che chiunque altro possa accedere al file.

Questo capitolo cita numerosi modi per proteggere l'utente ed i mini floppy da vari eventi indesiderabili. Probabilmente si troverà utile una o più di queste tecniche in un momento o in un altro. Si consideri innazitutto la preparazione di un apposito sistema "di lasciapassare".

Creazione di un sistema chiavi in mano

Si supponga che un medico voglia eseguire sur Apple II la contabilità del suo studio.
Idealmente, il personale dello studio dovrebbe essere in grado di accedere semplicemente ad Apple II, per trovarsi immediatamente inserito nel programma contabile del medico. Dato che il programma contabile comunica (almeno si spera) con l'utente nella normale lingua italiana, il personale non ha bisogno di conoscere il BASIC o qualsiasi altra cosa su Apple II.

Il computer diventa un sistema contabile e le sue caratteristiche interne diventano poco importanti, dato che tutto il personale deve sapere soltanto come usare il programma di contabilità.

Questa è l'essenza di un sistema "chiavi in mano" : dal punto di vista dell'utente, il computer è un dispositivo che esegue soltanto un compito particolare e mettere in funzione il sistema è altrettanto semplice quanto inserire una chiave in una serratura. In questo caso, la "chiave" consiste semplicemente nel manovrare l'interruttore di Apple ed inserire il corretto mini floppy. Non occorre esperienza in computer per poter far ciò.

È possibile usare il programma greeting, denominato quando è stato inizializzato il mini floppy per trasformare Apple II in un sistema chiavi in mano. Si supponga di volere che il computer esegua il programma COLOR DEMO (fornito sul mini floppy System Master) ogniqualvolta si lancia Disk II. Ecco come procedere :

1) Inizializzare un mini floppy vuoto (INIT) come descritto nel Capitolo 2.

2) Disporre il mini floppy System Master nell'unità disco e battere RUN COLOR DEMO Una volta verificata la corretta esecuzione del programma, battere

   {CTRL}C
   per interrompere il programma e ritornare al BASIC.

3) Inserire il mini floppy recentemente inizializzato nell'unità disco. Si suppone che il programma greeting sia stato chiamato HELLO quando il mini floppy è stato inizializzato.

4) Il programma COLOR DEMO è ora in memoria. Quando si batte

   SAVE HELLO

   il DOS cancella il programma originale greeting denominato HELLO e memorizza il programma COLOR DEMO sotto il nome file HELLO.

Il programma COLOR DEMO è ora il programma greeting sul mini floppy dell'utente. Per controllare che tutto funzioni come previsto, lanciare il disco (PR # 6). Si dovrebbe ottenere lo stesso programma usato nella fase 2).

È stato così creato un sistema chiavi in mano : ogniqualvolta il mini floppy viene lanciato, automaticamente carica (LOAD) il programma COLOR DEMO e lo esegue (RUN).

LOCK e UNLOCK (blocca e sblocca)

Talvolta si desidera impedire che in un particolare programma venga accidentalmente cancellato da un mini floppy : a ciò provvede il comando LOCK.

Esempio :

```
LOCK NESS, D2
```

Il CATALOG del contenuto di questo mini floppy presenta ora un asterisco (*) vicino all'entrata per NESS.

Se si decide di non conservare più questo file bloccato (LOCK), il comando UNLOCK sblocca il file.

Esempio :

```
UNLOCK NESS
```

La sintassi per il commando è :

```
    LOCK f [,Ss] [,Dd] [,Vv]
UNLOCK f [,Ss] [,Dd] [,Vv]
```

L'interpretazione della notazione è discussa nella parte sintassi del Capitolo 3.

Se si tenta di cancellare (DELETE) o ridenominare (RENAME) un file bloccato (LOCK) si riceve il messaggio

```
FILE LOCKED
```

Questo messaggio compare anche se si tenta di memorizzare (SAVE) un file usando il nome di un file bloccato (se il file che si sta cercando di memorizzare è nello stesso linguaggio del file bloccato).



Se si tenta di memorizzare (SAVE) un file usando il nome di un file bloccato in un diverso linguaggio, compare il messaggio

```
FILE TYPE MISMATCH
```

Provare di nuovo usando un diverso nome file.

Verify (verifica)

Di tanto in tanto talune informazioni possono non venir registrate correttamente su un mini floppy. Ciò può verificarsi se il mini floppy è graffiato o sporco. Ad esempio il comando VERIFY segnala un file che può essere danneggiato o scritto scorrettamente.

La sintassi è quella solita per i comandi DOS :

```
VERIFY f [,Ss] [,Dd] [,Vv]
```

Seguono esempi del modo in cui usare il comando

```
VERIFY SAM
VERIFY FINANCE-8,D2,V22
```

VERIFY controlla se le informazioni nel file specificato sono coerenti. In caso positivo, non compare alcun messaggio : viene semplicemente presentato il carattere di richiesta per il linguaggio in uso :

- > per Integer BASIC
-] per Applesof
- * per Monitor

Per contro VERIFY non controlla se un programma è rovinato e meno. Se è stato memorizzato (SAVE) un programma che era stato in qualche modo danneggiato, questo rimane **ancora** danneggiato sul mini floppy.
Se il comando VERIFY trova un errore, compare il messaggio

```
I/O ERROR
```

Se si tenta di verificare (VERIFY) un file che non è su disco, viene presentato il messaggio

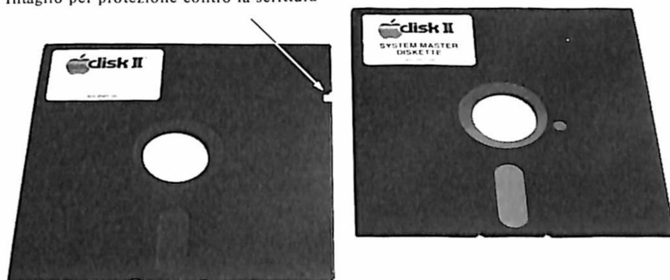
```
FILE NOT FOUND
```

È possibile usare VERIFY da Integer BASIC, Applesoft o Monitor. Da questi linguaggi è possibile anche VERIFY qualsiasi tipo di file compresi i file di testo (vedere Capitoli 6, 7 e 8) ed i programmi in linguaggio macchina (vedere Capitolo 9).

Protezione di un disco dalla scrittura

Il comando LOCK consente di proteggere un particolare file, ma talvolta occorre essere sicuri che **tutti** i files su un determinato mini floppy non siano accidentalmente cancellati da una sovrascrittura e quindi persi. Per "proteggere dalla scrittura" un mini floppy, occorre semplicemente coprire l'intaglio di protezione contro la scrittura di forma quadrata che si trova sul fianco del disco. A questo scopo, sono fornite etichette autoadesive quando si acquistano i mini floppy. In mancanza di esse, qualsiasi pezzo di nastro robusto può andare bene. Notare che il System Master non ha l'intaglio per la protezione contro la scrittura : esso è **permanentemente** protetto.

Intaglio per protezione contro la scrittura



Se si decide di riutilizzare un mini floppy protetto contro la scrittura, basta rimuovere l'etichetta (detta spesso "linguetta") che copre l'intaglio di protezione.

Alcuni programmi non possono essere usati in un mini floppy protetto contro la scrittura. Un esempio di tale programma è ANIMALS, uno dei programmi di dimostrazione del System Master. Inserire il mini floppy System Master nell'unità disco, e lanciare il DOS se occorre farlo.
Ora battere

```
LOAD ANIMALS
```

che mette il programma in memoria. Battere ora

```
RUN
```

dopo di che compare il messaggio

```
WRITE PROTECTED
STOPPED AT 1Ø4Ø
```

Il programma ANIMALS non viene eseguito su un mini floppy protetto contro la scrittura in quanto memorizza informazioni sul mini floppy ogniqualvolta si esegue il gioco. Quando si esegue il programma (RUN), il mini floppy nell'unità disco **non deve** essere protetto in scrittura, altrimenti non è possibile scrivere le informazioni.

Ora ANIMALS è in memoria, ma non è possibile eseguirlo con il mini floppy System Master. Inserire un mini floppy inizializzato, cioè non protetto, nell'unità disco.
Quindi battere

```
RUN
```

Ora è possibile giocare con ANIMALS, un gioco che "ricorda" ciò che gli si "insegna" memorizzando le informazioni sul mini floppy. Al termine del gioco, battere

```
SAVE ANIMALS
```

in modo da avere il gioco su un mini floppy non protetto contro la scrittura.

Se si batte

```
CATALOG
```

Si può vedere che sul mini floppy c'è non solo una copia di ANIMALS ma anche un nuovo file denominato ANIMALS-FILE che è stato creato dal programma ANIMALS.

Protezione dell'utente dai disastri

I mini floppy sono robusti ed affidabili rispetto ad alcuni altri supporti per memorizzare programmi e dati. Ma è sempre possibile perdere o distruggere tutte le informazioni su un mini floppy. Il mini floppy può venire graffiato o danneggiato dal calore; può andare perso oppure può essere morsicato da un cane; qualcuno può decidere di usarlo come frisbee alla spiaggia; se un mini floppy non è protetto contro la scrittura, può essere accidentalmente cancellato. Ed un mini floppy può anche consumarsi – la durata media è di circa 40 ore, di funzionamento continuo.

** MORALE **

Tenere a disposizione più di una copia di un programma se non si vuole perderlo, eseguite cioè delle copie di "back-up".

Se si è nel mezzo della compilazione o della modifica di un programma, un modo per creare il programma di riserva consiste nel tenere copia delle precedenti versioni : se la versione corrente va persa, si può riandare alla versione precedente e sperare di non perdere troppo tempo in programmazione.

Un modo ottimo consiste nel terminare ciascun nome file con un numero che cambia da versione a versione. Ad esempio, si supponga di iniziare a scrivere un programma denominato FINANCE. La prima volta che si memorizza il programma, lo si chiama FINANCE-1. La successiva volta in cui si lavora allo stesso programma lo si memorizza sotto il nome FINANCE-2; la terza volta, come FINANCE-3; e così via. Finirà con un'intera collezione di programmi FINANCE, con il numero di versione più alto che rappresenta la versione più recente del programma.

È una buona idea memorizzare (SAVE) periodicamente un programma in via di sviluppo (con un nuovo numero di versione). Così facendo ogni 15 o 20 minuti, un'improvvisa mancanza di corrente o altro disastro non cancella **tutto** il lavoro. È possibile naturalmente, continuare immediatamente a lavorare dopo aver memorizzato la situazione corrente del programma – tanto per essere sicuri di assegnare un nuovo numero di versione per il successivo SAVE.Se il mini floppy incomincia a riempirsi, cancellare (DELETE) una parte delle versioni precedenti. Ma è una buona idea tenere a portata di mano parecchie versioni nel caso in cui accada qualcosa di calamitoso alla versione corrente. Oppure può soltanto essere necessario ricuperare una precedente versione – non tutte le revisioni sono in effetti dei miglioramenti.

La frase "backing" (creare la riserva) è anche usata per descrivere la conservazione di copie multiple di programmi su mini floppy separati. Ci sono due approcci per operare in questo modo. Il primo metodo usa una sola unità disco : basta memorizzare (SAVE) il programma su un mini floppy, rimuovere quel mini floppy dall'unità, inserire un altro mini floppy e SAVE il programma di nuovo.

Il secondo approccio comporta la duplicazione di tutte le informazioni da un mini floppy su un secondo mini floppy. Dettagli su questo approccio vengono discussi nel prossimo paragrafo.

Uso del programma di copiatura

Per copiare un intero mini floppy, su un altro di "back up", potete usare il programma COPY, residente nel System Master Diskette. Se state usando l'Applesoft BASIC, eseguite le copie con il programma COPYA.

In questi programmi di copia il mini floppy dal quale produrre la copia è denominato "original", ed il mini floppy sul quale viene eseguita la copia è detto "duplicate". Il mini floppy "duplicate" non deve necessariamente essere inizializzato (INIT) prima di effettuare la copia, ed in ogni caso dopo la copia, tutte le precedenti informazioni residenti

sul mini floppy "duplicate" saranno cancellate. Prima di iniziare la copia, è buona idea proteggere il mini floppy "original" contro la scrittura, cio evita di cancellarne accidentalmente il contenuto, anche se lo si inserisce nell'unità sbagliata.

I programmi presumono che il mini floppy "original" venga disposto nell'unità correntemente selezionata (l'unità dalla quale è stato caricato il COPY o COPYA), collegata all'unità di controllo nello slot correntemente selezionato. Per usare il numero di slot o di unità presunto per il mini floppy "original" basta premere il tasto RETURN quando il programma si attende che l'utente batta un numero. Se uno o l'altro valore presunto è sbagliato per il mini floppy "originale" nel sistema, si deve premere il numero corretto quando ciò viene richiesto dal programma.

Ecco un esempio dell'uso del programma COPY con i numeri presunti di slot e di unità. Esso presume che sia collegato un disk drive nello stesso slot dal quale è stato lanciato il COPY.

1) Disporre il mini floppy System Master nell'unità corrente. Battere

   RUN COPY

   e si vedrà comparire il messaggio

   APPLE DISKETTE DUPLICATION PROGRAM

   ORIGINAL SLOT : DEFAULT =6

2) Premere ora il tasto RETURN per indicare che si desidera usare il numero di slot presunto, slot 6 nel nostro esempio, per il mini floppy originale.

3) Quando si vede comparire il messaggio

   DRIVE : DEFAULT=1

   premere di nuovo il tasto **RETURN** per indicare che si desidera il numero di unità presunto, unità 1 nel nostro esempio, per il mini floppy originale.

4. Si vedrà ora comparire:

   DUPLICATE SLOT : DEFAULT=6

   e si risponderà RETURN

5) Quando verrà visualizzato il valore proposto per il numero unità.

   DRIVE : DEFAULT=2

   PREMERE 1 e poi RETURN.

6) Apparirà il messaggio:

   — PRESS 'RETURN' KEY TO BEGIN COPY —

   Questo è il segnale per rimuovere il System Master dall'unità e inserire il mini floppy originale, dal quale effettuare la copia.

7) Alla pressione del tasto Return, il programma procederà indicando di inserire il mini floppy originale, poi informando sull'attività di lettura in corso. In seguito vi indicherà quando sostituire il mini floppy "original" con il "duplicate" e quando sta procedendo alla inizializzazione e scrittura. L'operazione verrà ripetuta più volte fino al completamento della copia.

8) Quando la copia sarà terminata verrà chiesto se ne desiderate un'altra :

DO YOU WISH TO MAKE ANOTHER COPY?

Se rispondete Y (Yes), verrà ripetuta la procedura di copia, con le stesse locazioni per i mini floppy "original" e "duplicate". Siate sicuri di inserire l'originale quando risponderete Return per iniziare la copia :

— PRESS 'RETURN' KEY TO BEGIN COPY —

e usate un altro dischetto (nuovo o con informazioni obsolete) per produrre la copia.

Se non desiderate altre copie rispondete N (No) alla relativa domanda. In tal caso potrete utilizzare normalmente il sistema tenendo presente che il DOS considera unità proposta, quella relativa alla precedente locazione del disco "duplicate".

Se possedete più di una unità Disk, specificate diverse locazioni per mini floppy "original" e "duplicate", ed inserite i dischi prima di iniziare la copia. Nota : Con più di una unità Disk, dal sistema non verrà richiesto di scambiare continuamente i mini floppy, ma la copia procederà automaticamente.



Se cercate di effettuare una copia su un mini floppy protetto da scrittura otterrete il messaggio

I/O ERROR
STOPPED AT (un numero di linea)

Finchè non toglierete l'etichetta di protezione da scrittura, non protrete utilizzare il mini floppy per produrre delle copie.



Se lo sportellino dell'unità è aperto o manca il mini floppy otterrete uno dei messagi : I/O ERROR o UNABLE TO READ. Tali messaggi possono anche indicare qualche altro problema relativo ai diskette nei drives. Dopo tali messaggi siate sicuri, rilanciando il programma, delle locazioni "original" e "duplicate" poichè potrebbero essere cambiate.

CAPITOLO 5

Altre informazioni di "preparazione"

Debugging : MON e NOMON

Il procedimento col quale si cerca di far girare un programma nel modo desiderato è detto "debugging" e gli errori di programma sono spesso detti quindi "bugs". Tutti i comandi del disco e tutte le informazioni scambiate tra il computer e il disco non sono normalmente presentati sullo schermo, ma quando si esegue il debugging, il controllo di queste informazioni può aiutare ad individuare i vari problemi.

Il comando MON consente di controllare (MONitor) numerose informazioni. Per escludere di nuovo le varie parti dallo schermo, usare il comando NOMON (NO MONitor).

In questi comandi possono essere usati tre diversi parametri :

C sta per comandi al disco (tipo ad esempio OPEN, READ, ecc.)
I sta per input dal disco (quando si legge un file, READ)
O sta per output verso il disco (quando si scrive su un file, WRITE).

Questi parametri sono usati soltanto con i comandi NOMON e MON. Solitamente è in atto NOMON C,I,O : non avviene cioè alcun controllo.

La sintassi per i comandi è la seguente :

```
MON    [C]  [,I]  [,O]
NOMON  [C]  [,I]  [,O]
```

Con i comandi NOMON e MON deve essere presente almeno uno dei tre parametri, altrimenti il comando sarà ignorato. I parametri possono comparire in qualsiasi ordine e come al solito devono essere separati da virgole.

Ci sono sette diversi modi in cui può essere usato il comando MON :

comando	cosa controlla
MON C	Comandi al disco
MON I	Input al disco
MON O	Output verso il disco
MON I,O	Input da e output verso il disco
MON C,I	Comandi a, e input dal disco
MON C,O	Comandi a, e output verso il disco
MON C,I,O	Comandi a, input da e output verso il disco

*** **NOTA** ***

Un comando MON rimane in atto fino a che non incontra un comando NOMON, INT o FP (solo in firmware)
/oppure/ fino a che non si rilancia il sistema
/oppure/ non si esegue una ripartenza (3DØG), da Monitor.

oppure
Si preme RESET

Un piccolo trucco : è possibile emettere un comando MON e successivamente cancellarlo senza intervenire sul formato dello schermo – anche il nomon non compare sullo schemo.

Si supponga di eseguire un commando MON, vale a dire

MON C,I,O.

Per cancellare il comando senza farlo comparire sullo schermo, comprendervi

PRINT D$; ''NOMON C,I,O''; VTAB PEEK (37) : CALL –868

dove D$, come solito, contiene CTRL-D.

MAXFILES

Il DOS consente un massimo di 16 files attivi (in uso) contemporaneamente. DOS tratta parecchi tipi di files in aggiunta ai files di programma BASIC finora discussi. Vedere il Capitolo 6 per una discussione dei files di testo sequenziale, il Capitolo 8 per i files di testo ad accesso casuale, il Capitolo 9 per i comandi DOS usati con i files binari (in linguaggio macchina).

Il comando MAXFILES specifica quanti sono i files attivi permessi. Quando si lancia il DOS, viene eseguito il comando

MAXFILES 3

che definisce la condizione presente : possono essere attivi simultaneamente fino a 3 files, fino a che non viene eseguito un altro comando MAXFILES.

La sintassi del comando 1

MAXFILES n

dove n deve essere un intero da 1 a 16. La specifica di un valore al di fuori di questo campo fa comparire un messaggio
SYNTAX ERROR da APPLESOFT o da Integer BASIC ; dal Monitor, un beep è la sola indicazione che è stato fatto qualcosa di sbagliato.

Per ciascun file specificato, MAXFILES accantona 595 bytes di spazio di memoria detto **buffer del file**. Questo ulteriore spazio di memoria per ciascun file attivo viene usato per aiutare a tener conto del fatto che la velocità di memoria è di gran lunga maggiore della velocità di accesso al disco, che comporta un movimento meccanico – la testina del disco deve in effetti esplorare il mini floppy. Pertanto, in nome dell'efficienza, il buffer dei files è usato per "bufferizzare" le informazioni che vanno da e verso il mini floppy.

Se si richiamano le informazioni da un mini floppy, il DOS introduce 256 caratteri alla volta e li mette nella parte "input" del buffer dei files, **quindi**, li restituisce, qualunque sia il subset di questi 256 caratteri richiesto dal programma. Se si inviano informazioni a un mini floppy, i caratteri sono caricati nella parte "output" del buffer dei files, fino a che non si sono accumulati 256 caratteri, che vengono successivamente inviati tutti in una volta.

Si supponga di avere MAXFILE 1 e un file è attivo. Il tentativo di eseguire un comando DOS (tipo ad esempio CATALOG) provoca la comparsa del messaggio

NO BUFFERS AVAILABLE

Quando il sistema è lanciato, il numero dei files attivi (n) è presunto 3, per cui sono riservati 1785 bytes di memoria per 3 buffer dei files. Nella maggior parte delle circostanze, non occorono però più di tre files attivi. Se ne occorrono di più, battere

MAXFILE n

(dove n è il numero dei files richiesti) nel modo ad esecuzione immediata indicare MAXFILE prima di caricare ed eseguire un programma.



Nel modo ad esecuzione immediata, l'aumento di MAXFILES cancella i programmi Integer BASIC e confonde le stringhe APPLESOFT, dato che HIMEM : è spostato verso il basso senza spostare il programma o le stringhe. Per evitare questo problema occore ripristinare MAXFILES prima di caricare ed eseguire un programma.



Se MAXFILES viene usato all'interno di un programma, cambia i puntatori della memoria ed è possibile che vada perso un GOTO, GOSUB o qualche altra istruzione. Se si deve cambiare MAXFILES dall'interno di un programma APPLESOFT occorre far sì che il comando MAXFILES sia la prima istruzione del programma stesso, prima che venga dichiarata qualsiasi variabile a stringa.
Ad esempio

```
1Ø PRINT CHR$(4); ''MAXFILES 5''
```

Per usare MAXFILES dall'interno di un programma Integer BASIC occorre creare un file EXEC, come discusso alla fine del capitolo 6.

TRACE

Il comando di Applesoft TRACE è un utile strumento di debugging. Ma quando è in atto TRACE, i comandi DOS all'interno dei programmi Applesoft non funzionano, in quanto TRACE stampa il numero di riga senza **RETURN** prima del comando DOS. C'è una parziale soluzione al problema : è cioè possibile inserire un **RETURN** (CHR$(13)) nella stringa D$:

```
1Ø D$=CHR$(13) + CHR$(4)
```

e quindi la maggior parte dei comandi DOS ,funzionerà correttamente anche se è in atto TRACE.

Se il comando TRACE è attivo, ed il DOS cerca di leggere dati da un file, stranamente questi dati vengono richiesti da tastiera. Ciò limita l'utilità del TRACE in ambiente DOS.

Uso del programma di aggiornamento "MASTER CREATE"

Come discusso nel Capitolo 2, INIT viene usato per creare mini floppy secondari. In questo Capitolo si imparerà invece a creare mini floppy principali. La distinzione tra un secondario e un principale non è molto evidente: entrambi vengono deliziosamente presentati nella plastica nera più alla moda (no, non in pelle nera!). Tocca quindi all'operatore visionare il programma "greeting" e l'etichetta del mini floppy per stabilire qual'è il secondario e qual'è il principale.
Il mini floppy System Master contiene un programma denominato MASTER CREATE che può venir eseguito su Apple II con almeno 16K di memoria. Il programma MASTER CREATE esegue quanto segue:

* Converte un disco secondario (il cui DOS dipende da dimensioni di memoria) in uno principale (il cui DOS è auto rilocante, in modo da utilizzare in modo efficiente qualsiasi tipo di sistema).
* Dà al mini floppy aggiornato un nuovo nome di programma "greeting", il nome che DOS tenterà di eseguire (**RUN**) ogniqualvolta il mini floppy è lanciato.

Il programma MASTER CREATE deve essere usato con un mini floppy che sia già stato inizializzato (INIT) e non funziona con un mini floppy che è protetto in scrittura.

Ecco un esempio di come aggiornare (UPDATE) il mini floppy inizializzato (INIT) nel Capitolo 2 (quello contenente il programma ONE TO TEN) per convertire il disco secondario creato da INIT in un mini floppy principale. Per comodità, a quel mini floppy si farà riferimento come mini floppy uno nella discussione che segue.

Prima di usare il MASTER CREATE eseguire quanto segue:

1. Inserire il disco che si desidera aggiornare – mini floppy 1 in quest'esempio – nell'unità disco ed eseguire il programma "greeting" – denominato HELLO sul mini floppy 1. Il messaggio presentato da un programma "greeting" dovrebbe comprendere la versione del DOS usato per inizializzare il mini floppy, e la sua condizione cioè secondario o principale.

2. Cambiare le righe appropriate del programma "greeting" per presentare la nuova informazione "MASTER DISKETTE". Quindi, memorizzare (SAVE) questa nuova del programma "greeting". Se l'etichetta esterna del mini floppy richiede un analogo cambiamento, apportarlo ora.

3. Annotare il nome del programma "greeting". Se si desidera che il mini floppy aggiornato esegua **(RUN)** questo programma ogni volta che viene lanciato, esattamente come avveniva prima dell'aggiornamento, occorre dare successivamente questo nome di programma "greeting" al programma MASTER CREATE. Se si desidera che il programma "greeting" abbia qualche altro nome diverso da quello attuale, occorre ridenominare in questo momento il programma "greeting" (RENAME). Successivamente, occorrerà dare il nuovo nome al programma MASTER CREATE.

Per usare il MASTER CREATE, eseguire quanto segue:

4. Inserire il mini floppy System Master nell'unità disco, lanciare il DOS e da entrambi i tipi BASIC scrivere BRUN MASTER CREATE.

5. Si dovrebbe osservare il messaggio

DOS 3.3 MASTER-CREATE UTILITY
COPYRIGHT 198Ø BY APPLE COMPUTER INC
ALL RIGHTS RESERVED

(NOW LOADING DOS IMAGE)

6. Verrà quindi detto di battere il nome del programma "greeting" da usare dal mini floppy aggiornato:

PLEASE INPUT THE GREETING PROGRAM'S FILE NAME :

Supponiamo che quando è stato memorizzato il programma "greeting" revisionato (SAVE) sul disco 1 (fase 3 precedente), sia stato usato il nome HELLO. Battere quindi

HELLO

a meno che non si desideri che il disco esegua (**RUN**) qualche altro nome di programma ogniqualvolta viene lanciato. Quando si preme RETURN per impostare il nome del programma "greeting" si vedrà comparire il messaggio:

```
REMEMBER THAT MASTER DOES NOT CREATE
THE GREETING PROGRAM, OR PLACE IT IN
THE DISK DIRECTORY
  THIS IS THE FILE NAME THAT WILL BE
PLACED WITHIN THE IMAGE :
      HELLO
  PLACE THE DISKETTE TO BE MASTERED IN
THE DISK DRIVE.
  PRESS <ESC> WHEN READY
NOTE : IF YOU WANT A DIFFERENT FILE NAME,
PRESS <ESC>
```

4. Seguire le istruzioni. Rimuovere il disco System Master dall'unità disco e sostituirlo con il disco che si desidera aggiornare – disco 1 in questo caso. Infine, premere il tasto RETURN per iniziare l'aggiornamento; il programma informerà l'utente quando il processo è completato.

5. Dopo aver usato il programma MASTER CREATE, occorre sempre rilanciare il DOS prima di eseguire qualsiasi altro lavoro.

*** **NOTA** ***

Il nome del programma "greeting" che è stato dato al programma MASTER CREATE non è inserito nel catalogo del disco. Esso dice soltanto al DOS del disco quale nome programma eseguire (**RUN**) ogniqualvolta il disco viene lanciato. Occorre però assicurarsi che il catalogo del disco contenga effettivamente un programma che abbia lo stesso nome attribuito dal programma MASTER CREATE.

Se ci si è dimenticati di farlo (saltando la fase 3 che precede), si vedrà comparire il messagio

FILE NOT FOUND

ogniqualvolta si lancia il disco.

*** **PROMEMORIA** ***

Occorre **ricordare** il nome del programma "greeting" per ciascun mini floppy.
Si può rendere la cosa molto semplice usando lo stesso nome di programma "greeting" su tutti i dischi.

CAPITOLO 6

Uso dei files sequenziali

Files di testo: introduzione

Talvolta si desidera usare il disco per caricare informazioni diverse dai programmi. È possibile, ad esempio, tenere copia della corrispondenza, un elenco di parole usate negli indovinelli, i risultati intermedi di un calcolo, oppure un elenco di distribuzione. Un file di testo, talvolta detto anche **file di dati**, consente di fare questo ed altro. Nell'elenco CATALOG la lettera T contrassegna i files di testo.

I files di testo sono creati e richiamati usando i comandi DOS in un programma Integer BASIC o Applesoft. Un file di testo può essere creato usando un programma scritto in un linguaggio e richiamato da uno che usa un programma scritto in un altro linguaggio.

La maggior parte dei programmi-esempio di questo manuale sono in Applesoft. Se si desidera convertire i programmi in Integer BASIC, occorre ricordare che in Integer BASIC non è possibile usare matrici di stringhe, per cui occorre dimensionare le variabili stringa (DIM). In un comando Integer BASIC, del tipo.

```
INPUT A$, B$, C$
```

solo **RETURN** (e non le virgole) possono separare le tre risposte. Questo manuale non insegnerà a far girare ciascun programma in Integer BASIC: vedere per questo l'Appendice M del **Manuale di riferimento del BASIC Applesoft** per i particolari di conversione tra i linguaggi. Per alcuni suggerimenti sul cambiamento del BASIC in cui gira il programma, dopo che il programma è stato scritto, vedere pagina 76 di questo Manuale DOS.

I commandi DOS, LOAD e **RUN** (e anche BLOAD e BRUN) non possono essere usati con un file di testo. Un tentativo di farlo fa comparire il messagio

```
FILE TYPE MISMATCH
```

LOAD e **RUN** si aspettano un file di programma BASIC (e BLOAD e BRUN si aspettano un file di linguaggio macchina binario) e non un file di testo. Per contro, occorre scrivere programmi che inviino dati ad file di testo e richiamino dati da un file testo, usando i comandi DOS discussi in questo capitolo:

```
OPEN
CLOSE
READ
WRITE
APPEND
POSITION
EXEC
```

I comandi OPEN, READ, WRITE, APPEND e POSITION non possono essere usati nel modo ad esecuzione immediata. Se si tenta di farlo, compare il messaggio

```
NOT DIRECT COMMAND
```

Questi comandi devono essere usati nel modo ad esecuzione differita, cioè dall'interno di un programma. I comandi CLOSE ed EXEC possono essere usati nel modo ad esecuzione immediata.

Oltre ai comandi sopra citati, anche i comandi DOS

```
LOCK e UNLOCK
DELETE
```

MON e NOMON
VERIFY
CATALOG

funzionano con i files di testo allo stesso modo in cui lavorano con i files di programma.

Ci sono due diversi tipi di files di testo: files di testo sequenziali e file di testo ad accesso casuale. Entrambi i tipi memorizzano stringhe di codici ASCII per rappresentare i dati, ma in formati diversi. Qui di seguito sono indicati i diagrammi dei due tipi di files di testo (il carattere "↵" rappresenta il carattere **RETURN**, inviato automaticamente al termine dalla maggior parte delle istruzioni PRINT).

UN FILE DI TESTO SEQUENZIALE

Carattere:	7	↵	A	T	↵	O	N	E	↵	B	L	O	W	↵							
ASCII:	55	13	65	84	13	79	78	69	13	66	76	79	87	13	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ	ØØ
Byte del file:	Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15	16	17	18	19	2Ø
Campo:	Ø		1			2				3											

UN FILE DI TESTO AD ACCESSO CASUALE
(Esempio: lunghezza del record 5, un campo per record)

Carattere:	7	↵				A	T	↵			O	N	E	↵		B	L	O	W	↵	
ASCII:	55	13	ØØ	ØØ	ØØ	65	84	13	ØØ	ØØ	79	78	69	13	ØØ	66	76	79	87	13	ØØ
Byte del file:	Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15	16	17	18	19	2Ø
Byte del record:	Ø	1	2	3	4	Ø	1	2	3	4	Ø	1	2	3	4	Ø	1	2	3	4	Ø
Campo:	Ø					[Ø]					[Ø]					[Ø]					
Record:	Ø					[1]					[2]					[3]					

I termini "campo" e "record" verranno discussi nei Capitoli 6, 7 e 8. I comandi OPEN, CLOSE, READ, WRITE e POSITION sono usati con entrambi i files, ma in modo alquanto diverso. Sotto alcuni aspetti, i files di testo sequenziali sono più semplici da usare e da comprendere, per cui verranno discussi per primi l'uso e la struttura dei files di testo sequenziale. L'uso di files di testo ad accesso casuale è descritto nel Capitolo 8. Informazioni tecniche più dettagliate su tutti i tipi di files si trovano nell'Appendice C.

Files di testo sequenziali: alcuni esempi

Si supponga di voler creare un file contenente un elenco di parole da usarsi in un gioco di indovinelli. Qui ci sono due coppie di programmi che trattano con tale file. Il primo programma in ciascuna coppia crea un file di testo su un mini floppy.
Il secondo programma in ciascuna coppia richiama i dati caricati nel file di testo dal mini floppy. Questo programma crea un file di testo denominato WORDS1, contenente le parole APPLE, BANANA, CATALOG, DORMANT, EAGLE, FRUIT, GOOSE, HAT e ICICLE.

```
 1Ø  REM MAKE WORDS1
 2Ø D$ = " ": REM  CTRL-D
 3Ø  PRINT D$; "OPEN WORDS1"
 4Ø  PRINT D$; "WRITE WORDS1"
 5Ø  PRINT "APPLE"
 6Ø  PRINT "BANANA"
 7Ø  PRINT "CATALOG"
 8Ø  PRINT "DORMANT"
 9Ø  PRINT "EAGLE"
1ØØ  PRINT "FRUIT"
11Ø  PRINT "GOOSE"
12Ø  PRINT "HAT"
13Ø  PRINT "ICICLE"
14Ø  PRINT D$; "CLOSE WORDS1"
15Ø  END
```

La riga 3Ø apre (OPEN) il file, usando il normale formato per inviare un comando DOS dall'interno di un programma BASIC. OPEN dispone un file di testo denominato WORDS1 nel CATALOG (se non c'era precedentemente).

Il comando WRITE della riga 4Ø provoca il successivo output dalle istruzioni PRINT da inviare al file di testo denominato, invece che inviarlo al video.

Così in questo programma, ciascuna istruzione PRINT nelle righe da 5Ø a 13Ø invierà la parola all'interno delle virgolette al file di testo WORLD1 e non al video.

La riga 14Ø chiude (CLOSE) e il file termina il processo di scrittura nel file stesso.

Se il programma viene eseguito (**RUN**) e non si è nel modo Monitor (MON) non si vedrà nulla: solitamente i comandi DOS e l'input e l'outpout sul disco non sono presentati. Ma se, come spiegato nel Capitolo 5, si è battuto:

MON C,I,0

(oppure più semplicemente

MON C, 0

dato che non è coinvolto alcun input dal disco) e quindi si esegue il sudetto programma (**RUN**), si vedrà quanto segue:

```
OPEN WORDS1
WRITE WORDS1
APPLE
BANANA
CATALOG
DORMANT
```

```
EAGLE
FRUIT
GOOSE
HAT
ICICLE
CLOSE WORDS1
```

A questo punto sul disco è disponibile un file denominato WORDS1. WORDS1 sarà contrassegnato con una "T" nel catalogo per indicare che si tratta di un file di testo. Il file si compone di elementi di dati (in questo caso parole) separati da vari **RETURN**. Un carattere **RETURN** viene automaticamente inviato al termine di ogni istruzione PRINT che non termina con una virgola o con un punto e virgola. Notare che in questo senso ciascun **RETURN** è un carattere anzichè una azione – in particolare è il carattere col codice 13 ASCII.

Ciascun elemento di dati, che termina con un proprio carattere **RETURN** è denominato campo. Un campo viene caricato nel file di testo sotto forma di una serie di caratteri rappresentati dai rispettivi codici ASCII. L'ultimo carattere in ciascum campo deve essere **RETURN**, codice ASCII 13.

WORDS1 è detto un file di testo sequenziale in quanto ciascun campo è caricato cominciando immediatamente dopo il carattere **RETURN** nel campo precedente. Quando caricati sul disco, i campi possono essere di diverse lunghezze: La parola APPLE richiede 6 bytes (1 per ciascuna lettera più 1 per il carattere **RETURN**), BANANA richiede 7 bytes e così via. Un file di testo sequenziale viene caricato sul disco sotto forma di una lunga e continua serie di caratteri codificati in ASCII, una catena di campi senza intervalli tra un campo e il successivo.

Una volta che WORDS1 è sul mini floppy, sorge immediatamente la domanda "com'è possible richiamarlo?". Il seguente programma Applesoft richiamerà WORDS1.

```
1Ø  REM RETRIEVE WORDS1
2Ø  D$ = " " : REM CTRL-D
3Ø  PRINT D$; "OPEN WORDS1"
4Ø  PRINT D$; "READ WORDS1"
5Ø  FOR I = 1 TO 9
6Ø  INPUT A$(1)
7Ø  NEXT I
8Ø  PRINT D$; "CLOSE WORDS1"
9Ø  END
```

La riga 3Ø apre il file (OPEN); la riga 4Ø dice al DOS che tutte le successive istruzioni INPUT o GET si riferiranno al file su disco dichiarato anzichè alla tastiera di Apple. È come se il disco battesse le risposte invece dell'operatore. Un comando INPUT provoca sempre la battitura su Apple di un campo completo, che termina con il rispettivo carattere **RETURN**. Se segue un altro comando input, questo provoca la lettura nel successivo campo e così via. Così le righe da 5Ø à 7Ø fanno sì che DOS parta all'inizio di WORDS1 e richiami 9 campi disposti nella matrice A$(1), A$(2), A$(3), A$(4),... A$(9). La riga 8Ø chiude (CLOSE) educatamente il file.

Se non è in atto MON C, I, O quando viene eseguito il suddetto programma (**RUN**), non si vedrà nulla sul video. Ma se è in atto MON C,I,O (oppure semplicemente MON C,I) si vedrà:

```
OPEN WORDS1
READ WORDS1
?APPLE
?BANANA
?CATALOG
?DORMANT
?EAGLE
?FRUIT
```

```
?GOOSE
?HAT
?ICICLE
CLOSE WORDS1
```

Prima di ciascun input dal disco compare un punto di domanda esattamente come davanti a ciascun normale input da tastiera.

Per controllare che tutto ha funzionato come desiderato, cercare di battere

```
PRINT A$(2), A$(9), A$(4)
```

si vedranno comparire le parole BANANA -- da A$(9) -- quindi ICICLE e infine DORMANT. Si tratta di un modo semplice per controllare che le informazioni siano state lette correttamente.



Se si modifica il programma MAKE WORDS1 per creare diverse parole, assicurarsi di cancellare WORDS1 (DELETE) prima di rieseguire (**RUN**) MAKE WORDS1. In caso contrario, si può finire con un miscuglio di parole vecchie e di nuove.

Ecco come creare un file sequenziale denominato WORDS2 contenente le stesse parole di WORDS1 ma con tutte le 9 parole in un solo campo. Ciascuna parola è seguita da una virgola cosìcchè può essere usata un'istruzione INPUT con variabili multiple (9, in questo caso) per richiamare le parole separate.

```
10  REM MAKE WORDS2
20  D$ = " " : REM CTRL-D
30  PRINT D$; "OPEN WORDS2"
40  PRINT D$; "WRITE WORDS2"
50  PRINT " APPLE, BANANA, CATALO
    G,";
60  PRINT " DORMANT, EAGLE, FRUIT
     ,";
70  PRINT " GOOSE, HAT, ICICLE"
80  PRINT D$, "CLOSE WORDS2"
90  END
```

Notare che il comando PRINT nella riga 50 termina con un punto e virgola. Un punto e virgola al termine di un comando PRINT interrompe la stampa automatica di un carattere **RETURN** dopo l'ultimo carattere dei dati. Pertanto, i caratteri inviati al disco dal successivo comando PRINT compariranno nello stesso scampo dei caratteri inviati dal comando PRINT della riga 50. Il comando PRINT della riga 60 termina a sua volta con un punto e virgola, cosìcchè il campo non ha il carattere **RETURN** che segna la fine. Il comando PRINT della riga 70 termina senza un punto e virgola, consentendo di inviare il carattere automatico finale **RETURN** per ultimo. Ciò termina il campo che ora contiene tutti i caratteri stampati (PRINT) dalle righe 50, 60, 70.



Le virgole in un comando PRINT non per DOS, solitamente inviano caratteri ai campi tabulari definiti sul video. Per contro, le virgole non servono a questa stessa funzione di formazione nei comandi PRINT usati quando si scrive sul disco (WRITE): queste virgole sono trattate come se fossero dei punti e virgola. Nella scrittura su disco, le voci separate da virgole verranno concatenate, senza la presenza di spazi intermedi. Una virgola al termine di un comando PRINT ha lo stesso effetto di un punto e virgola: non viene inviato carattere automatico finale **RETURN**.

Quando il programma MAKE WORDS2 viene eseguito (**RUN**) con MON C,I, O in atto si vede comparire:

```
OPEN WORDS2
WRITE WORDS 2
 APPLE, BANANA, CATALOG, DORMANT, EAGLE,
 FRUIT, GOOSE, HAT, ICICLE
CLOSE WORDS2
```

Questo programma Applesoft richiama WORDS2:

```
1Ø  REM RETRIEVE WORDS2
2Ø D$ = "": REM CTRL–D
3Ø  PRINT D$, "OPEN WORDS2"
4Ø  PRINT D$; "READ WORDS2"
5Ø  INPUT A1$, A2$, A3$, A4$, A5$, A6$
     , A7$, A8$, A9$
8Ø  PRINT D$; "CLOSE WORDS2"
9Ø  END
```

Quando il suddetto programma viene eseguito con MON C,I,O in atto, si vede comparire :

```
OPEN WORDS2
READ WORDS2
? APPLE, BANANA, CATALOG, DORMANT, EAGLE
, FRUIT, GOOSE, HAT, ICICLE
CLOSE WORDS2
```

In Integer BASIC, le virgole possono separare risposte di input multiple per variabili numeriche, ma non per le variabili a stringa. Soltanto i caratteri RETURN possono separare le risposte multiple quando viene usato INPUT con variabili a stringa multiple. Pertanto, in Integer BASIC, il programma RETRIEVE WORDS2 assegnerà l'intero campo (9 parole, 8 virgole e 6 spazi) alla variabile A1$ e pertanto si ottiene il messaggio END OF DATA quando non c'è alcun campo da assegnare a A2$.

In Applesoft BASIC, si può anche usare il commando GET per richiamare i dati da un file di testo, carattere per carattere. Ciò ha il vantaggio, ad esempio, di poter definire qualsiasi carattere per contrassegnare la fine di parole. Il seguente programma Applesoft richiama anche il file di testo WORDS2.

Nella riga 1Ø, il comando CLEAR definisce a zero tutte le variabili, (compresa I e tutti gli A$(I). La riga 2Ø usa il modo alternativo di Applesoft di definire D$ a **CTRL-D** (4 è il codice ASCII per **CTRL-D**). Questo metodo evita il carattere di controllo invisibile (e non copiabile).

```
1Ø  CLEAR : REM GET WORDS2
2Ø D$ = CHR$ (4) : REM CTRL–D
3Ø R$ = CHR$ (13) : REM RETURN
4Ø T$ = CHR$ (1) : REM CTRL–A
5Ø  PRINT D$, "OPEN WORDS2"
6Ø  PRINT D$, "READ WORDS2"
7Ø I = I + 1
8Ø  GET B$
9Ø  IF B$ = "," THEN GOTO 7Ø
1ØØ  IF B$ = R$ THE GOTO 13Ø
```

```
110 A$(I) = A$(I) + B$
115  PRINT T$;A$(I)
120  GOTO 80
130  PRINT R$; D$; "CLOSE WORDS2"
140  END
```

La riga 80 dà un carattere alla volta (GET) dal file di testo WORDS2, che era stato aperto (OPEN) per la lettura (READ) nelle righe 50 e 60. Se il nuovo carattere non è una virgola nè un **RETURN**, la riga 110 aggiunge il nuovo carattere alla fine della stringa A$(I). Quindi la riga 120 rimanda il programma alla riga 80 per ottenere (GET) il successivo carattere. Così il programma costruisce la prima parola, carattere per carattere, in A$(1).

Quando si trova una virgola, la prima parola termina, cosìcchè la riga 90 rimanda il programma alla riga 70 per incrementare I ed iniziare a raccogliere una nuova parola in A$(2). E cosi via. Infine, un carattere **RETURN** (R$) contrassegna la fine del campo cosìcchè la riga 100 riempie il programma alla riga 130 per chiudere (CLOSE) il file e terminare il programma. Notare l'uso di CHR$(13) nella riga 30. Non è possibile battere direttamente un carattere **RETURN** in una riga di programma BASIC (un **RETURN** termina una riga di programma) ma in Applesoft CHR$(13) è un carattere **RETURN**.

Quando GET ottiene i caratteri dal disco, questi non sono presentati sul video anche nel modo MON C,I,O.

La riga 115 è stata aggiunta per consentire di vedere le parole così come sono costruite, carattere per carattere.



Una volta che un comando GET di Applesoft riceve la sua risposta da un file di testo su un mini floppy, sorgono i seguenti problemi:

1. Con NOMON C,I,O, il primo carattere stampato (PRINT) dopo GET non compare sullo schermo.
2. Con MON C,I,O il primo carattere stampato (PRINT) dopo GET compare sullo schermo.
3. In entrambi i modi, se un comando DOS è il primo elemento stampato (PRINT) dopo GET, il comando DOS può non essere eseguito in quanto manca il necessario **RETURN** précedente.

Nel programma GET WORDS2, il carattere non stampante "a perdere" **CTRL-A** (T$) era stato posto davanti al primo carattere desiderato PRINT nella riga 115. Ciò riguarda i dei precedenti problemi 1 e 2. Per risolvere il problema 3, il carattere **RETURN** (R$) è stato disposto davanti al comando DOS stampato (PRINT) nella riga 130, più o meno come era stato fatto con TRACE (vedere pagina 44).
Quando questo programma viene eseguito (**RUN**) con MON C,I,O in atto, si vede quanto segue (ma presentato su una colonna e non su tre):

```
OPEN WORDS2
READ WORDS2

A
AP
APP
APPL
APPLE

B
BA
BAN
BANA
BANAN
BANANA

C
CA
CAT
CATA
CATAL
CATALO
CATALOG

D
DO
DOR
DORM
DORMA
DORMAN
DORMANT

E
EA
EAG
EAGL
EAGLE

F
FR
FRU
FRUI
FRUIT

G
GO
GOO
GOOS
GOOSE

H
HA
HAT

I
IC
ICI
ICIC
ICICL
ICICLE
CLOSE WORDS2
```

E infine qui c'è un programma Appelsoft che crea un file WORDS3, con 2 parole nel primo campo, 3 parole nel secondo e 4 parole nel terzo.

```
10  REM MAKE WORDS3
20 D$ = CHR$ (4) : REM CTRL–D
30  PRINT D$; "OPEN WORDS3"
40  PRINT D$; "WRITE WORDS3"
50  PRINT "APPLE, BANANA"
60  PRINT "CATALOG, DORMANT, EAGLE"

70  PRINT "FRUIT, GOOSE, HAT, ICICLE
    "
80  PRINT D$; "CLOSE WORDS3"
90  END
```

Il primo campo conterrà

    APPLE, BANANA

ed è lungo 13 bytes, uno per carattere (devono essere contate anche le virgole) più uno per il carattere **RETURN**. Il secondo campo

    CATALOG, DORMANT, EAGLE

è lungo 22 bytes: il terzo campo

    FRUIT, GOOSE, HAT, ICICLE

è lungo 23 bytes.

Quando eseguito (**RUN**) con MON C,I,O in atto, si vedrà comparire:

```
OPEN WORDS3
WRITE WORDS3
APPLE, BANANA
CATALOG, DORMANT,EAGLE
FRUIT, GOOSE, HAT, ICICLE
CLOSE WORDS3
```

Qui c'è un programma per richiamare WORDS3 :

```
10  REM RETRIEVE WORDS3 : A
20 D$ = CHR$ (4) : REM CTRL–D
30  PRINT D$; "OPEN WORDS3"
40  PRINT D$; "READ WORDS3"
50  INPUT R$, S$
60  INPUT T$, U$, V$
70  INPUT W$, X$, Y$, Z$
80  PRINT D$, "CLOSE WORDS3"
90  END
```

Quando eseguito con MON C,I,O in atto, si vedrà comparite quanto segue:

```
OPEN WORDS3
READ WORDS3
?APPLE, BANANA
?CATALOG, DORMANT, EAGLE
?FRUIT, GOOSE, HAT, ICICLE
CLOSE WORDS3
```

I programmi per leggere (READ) i files di testo sequenziali WORDS1, WORDS2, WORDS3 erano stati accuratamente studiati per leggere (READ) esattamente il numero corretto di campi e il numero corretto di elementi per ciascun campo. In generale, un programma per richiamare un file di testo deve essere costruito intorno ad un file specifico. Se si fa un errore, i risultati possono creare una certa confusione. Ad esempio, si consideri il seguente programma "Sbagliato" per richiamare le parole nel file di testo WORDS3.

```
10  REM RETRIEVE WORDS3:B
20 D$ = "": REM CTRL–D
30  PRINT D$; "OPEN WORDS3"
40  PRINT D$; "READ WORDS3"
50  INPUT R$, S$
60  INPUT 7$,U$, V$
70  INPUT W$, X$, Y$
80  PRINT D$; "CLOSE WORDS3"
90  END
```

Con MON C,I,O in atto ecco cosa si dovrebbe vedere eseguendo (**RUN**) il programma

```
OPEN WORDS3
READ WORDS3
?APPLE, BANANA
?CATALOG, DORMANT, EAGLE
?FRUIT, GOOSE, HAT, ICICLE
?EXTRA IGNORED
CLOSE WORDS3
```

Il comando INPUT nella riga 7Ø ha fatto sì che l'intero campo contenente

```
FRUIT, GOOSE, HAT, ICICLE
```

venisse letto (READ) in Apple. Le prime tre parole sono state assegnate alle variabili W$, X$ e Y$. Ma non c'è variabile corrispondente alla quarta risposta inserita (INPUT), ICICLE, per cui compare il messaggio

```
EXTRA IGNORED
```

e l'esecuzione continua.

Ecco un altro programma "Sbagliato" per leggere (READ) il file di testo WORDS3:

```
1Ø  REM RETRIEVE WORDS3 : C
2Ø D$ = "" : REM CTRL–D
3Ø  PRINT D$; "OPEN WORDS3"
4Ø  PRINT D$; "READ WORDS3"
5Ø  INPUT R$, S$
6Ø  INPUT T$, U$,V$, W$
7Ø  INPUT X$, Y$, Z$
8Ø  PRINT D$; "CLOSE WORDS3"
9Ø  END
```

E qui c'è un'esecuzione (**RUN**) del programma con MON C,I,O.

```
OPEN WORDS3
READ WORDS3
?APPLE, BANANA
?CATALOG, DORMANT, EAGLE
??FRUIT, GOOSE, HAT, ICICLE
?EXTRA IGNORED
?
END OF DATA
BREAK IN 7Ø
```

Questa volta, la riga 6Ø ha fatto sì che il campo

```
CATALOG, DORMANT, EAGLE
```

venisse letto (READ) in Apple. Le tre parole sono state assegnate alle variabili T$ e V$. Ma il comando di INPUT della riga 6Ø aspettava quattro risposte, per cui ha provocato la lettura del successivo campo completo (READ) in Apple:

```
FRUIT, GOOSE, HAT, ICICLE
```

La prima parola, FRUIT, è assegnata all'ultima variabile della riga 6Ø, W$. Non ci sono altre variabili con questo comando INPUT, per cui compare il messaggio

EXTRA IGNORED

e l'esecuzione continua.

Non ci sono altri campi nel file, cosìcchè il comando INPUT della riga 7Ø fa comparire il messagio

END OF DATA

e il programma si ferma.

In un successivo Capitolo è discussa una coppia più generica di programmi, MAKE TEXT e RETRIEVE TEXT. Essi mostrano come rendere un programma più adattabile ai diversi files di testo.

Apertura (Open) e chiusura (Close) di files sequenziali

I files di testo sequenziali devono essere usati quando le informazioni vanno richiamante in modo lineare dall'inizio alla fine e quando le informazioni non richiedono molto aggiornamento o revisione correnti. Ad esempio, un file sequenziale potrebbe essere usato per contenere i dati per un gioco di indovinelli come nei precedenti programmi esempio.

Per creare un file di testo sequenziale, vengono usati i comandi

OPEN
WRITE
PRINT
CLOSE

nell'ordine indicato (quantunque non necessariamente così uno dopo l'altro). Per richiamare un file di testo sequenziale, vengono usati i comandi

OPEN
READ
INPUT
CLOSE

di nuovo, non necessariamente l'uno dopo l'altro. Entrambe le procedure sono illustrate nel paragrafo precedente.

Ocorre un certo rituale prima e dopo la creazione (WRITE) di un file di testo sequenziale. Prima di usare il file occorre aprirlo (OPEN). Dopo averlo creato, occorre chiuderlo (CLOSE). Lo stesso vale quando si richiama (READ) un file di testo sequenziale: aprirlo (OPEN) prima di leggerlo (READ) e chiuderlo (CLOSE) quando la lettura è terminata.



I files che sono stati aperti devono essere chiusi. La mancata chiusura di un file che era stato aperto e sul quale si era scritto con un comando WRITE, può dar luogo alla perdita di dati

La sintassi per questi comandi è simile a quella degli altri comando DOS.
(Nota: OPEN e CLOSE sono anche usati con files ad accesso casuale – vedere il Capitolo 8).

OPEN f [,Ss] [,Dd] [,Vv]
CLOSE [f]

Esempi:

OPEN SESAME
OPEN SHOP, D2, S7
CLOSE
CLOSE MOUTHED
CLOSE WINDOW

OPEN accantona spazio di lavoro in Apple per il file f (per coloro che se ne intendono, OPEN assegna un buffer di files di 595 bytes per eseguire l'input e l'output di questo file) e predispone il sistema per leggere o scrivere dall'inizio del file. OPEN definisce anche la slot e numeri di unità da usarsi dal successivo comando WRITE (o READ).

Il comando CLOSE libera lo spazio di lavoro in Apple (disassegna cioè il buffer dei files associati con il file f). Se f non è specificato, tutti i files OPEN verranno chiusi, con l'eccezione di qualsiasi file in uso col comando EXEC. I files EXEC sono discussi più avanti nel Capitolo 7. Anche OPEN talvolta chiude (CLOSE): OPEN dapprima controlla se il file di cui trattasi è già aperto (OPEN); in caso affermativo lo ciude (CLOSE) prima di riaprirlo.

Notare che il comando CLOSE non ha parametri di unità o di slot.
Se si batte

CLOSE MYFILE

qualsiasi file denominato MYFILE verrà chiuso (CLOSE), indipendentemente dallo slot e dal numero di unità associato al file.
Naturalmente, il comando

CLOSE

chiude (CLOSE) tutti i files (salvo un files in corso di esecuzione – EXEC) su tutte le unità disco.

In varie circostanze, si può voler cancellare un file f che può esistere o meno. Ciò è particolarmente importante per evitare problemi di sovrascrittura di un vecchio file (a meno che non si sovrascriva l'intero vecchio file, parte del vecchio file rimarrà presente all'estremtà del nuovo file). Si supponga un gioco che crei e usi il file SCORES ogniqualvolta viene eseguito e si voglia che il programma cancelli ogni vecchio file mediante quel nome all'inizio di ciascun nuovo gioco.
Il comando

DELETE SCORES

fa comparire il messaggio di errore

FILE NOT FOUND

se il file non esiste e il programma s'interrompe. Ecco un modo rapido per cancellare qualsiasi file denominato SCORES e riaprirlo (OPEN) per nuovi dati, sia che il file esista o meno:

```
 5  REM SCORES DELETER
1Ø D$ ="''": REM  D$ IS CTRL-D
15  PRINT D$ ''OPEN SCORES''
2Ø  PRINT D$ ''DELETE SCORES''
25  PRINT D$,''OPEN SCORES''
3Ø  REM
                    REMAINDER OF PROGRAM
                         HERE
```

Scrittura (write) di files sequenziali

Ecco un altro programma che crea un file di testo sequenziale. Questo programma Applesoft crea un file di testo denominato SAMPLE che contiene tre stringhe e 1Ø numeri.
Il file SAMPLE può o può non esistere ogniqualvolta il programma viene eseguito (RUN): se esiste deve essere cancellato (DELETE) in modo da rimuovere i vecchi dati dal file. Se non esiste e si tenta di cancellarlo (DELETE) si riceve il messaggio:

```
FILE NOT FOUND
```

e il programma s'interrompe. Le righe 2Ø e 3Ø si occupano del problema. Se SAMPLE già esiste, la riga 2Ø lo apre (OPEN) e la riga 3Ø lo cancella (DELETE). Se SAMPLE non esiste, la riga 2Ø crea un file SAMPLE e la riga 3Ø lo cancella (DELETE). Quando la riga 4Ø viene eseguita, crea un nuovo file pulito SAMPLE, cosìcchè viene evitato il problema di mescolare i files.

```
  5  REM MAKE SAMPLE
 1Ø  D$=CHR$ (4): REM CTRL-D
 2Ø  PRINT D$; ''OPEN SAMPLE''
 3Ø  PRINT D$; ''DELETE SAMPLE''
 4Ø  PRINT D$; ''OPEN SAMPLE''
 5Ø  PRINT D$; ''WRITE SAMPLE''
 6Ø  PRINT ''HI HO'': ''HI HO'': PRINT ''HI HO''
 7Ø  PRINT ''OFF TO THE DISK WE GO''
 8Ø  FOR J=1 to 1Ø
 9Ø  PRINT J
1ØØ  NEXT J
11Ø  print d-; ''CLOSE SAMPLE''
12Ø  END
```

Ecco ciò che si vede sul video quando si esegue (RUN) questo programma, se è in atto MON C,I,O

```
OPEN SAMPLE
DELETE SAMPLE
OPEN SAMPLE
WRITE SAMPLE
HI HO
HI HO
OFF TO THE DISK WE GO
1
2
3
4
5
6
7
8
9
1Ø
CLOSE SAMPLE
```

Prima di scrivere un file (WRITE), questo deve essere aperto (OPEN) e occorre chiuderlo (CLOSE) (in silenzio, per favore) una volta finito. Entrambi i comando OPEN e WRITE devono fare riferimento allo stesso nome file. Una volta eseguito un comando WRITE, qualsiasi successivo comando PRINT invia tutti caratteri al mini floppy, anzichè al video. In un'istruzione PRINT un comando WRITE è cancellato dall'uso di qualsiasi altro comando DOS. Anche il comando DOS "vuoto" (ossia semplicemente **CTRL-D**) ottiene lo stesso effetto.



Anche un comando INPUT della forma

```
INPUT X$
```

cancella un comando WRITE, ma soltanto dopo aver memorizzato come ultimo carattere dei file di testo, il che il comando INPUT normalmente presenta sullo schermo.
Se viene usata la forma

```
INPUT "WHAT's YOUR NAME"; X$
```

WRITE è cancellato dopo che i caratteri nella stringa sono stati inviati al mini floppy.



Un messaggio di errore cancella un comando WRITE ma solo dopo che l'intero messaggio di errore è stato caricato come ultimo campo nel file di testo.

La sintassi per il comando WRITE quando usato con i files sequenziali è

WRITE f

(Nota: Write viene anche usato con i files ad accesso casuale, vedere Capitolo 8).

Esempi:

WRITE LETTER
WRITE RIGHT

Il programma esempio presentato all'inizio di questo Capitolo è una semplice illustrazione degli elementi fondamentali necessari per creare un file di testo. Un programma Applesoft più generico, denominato MAKE TEXT è sil disco System Master che viene fornito con l'unità disco.

MAKE TEXT consente di creare un file di testo sequenziale contenente fino a 1ØØ stringhe; ciascuna stringa può avere al massimo 239 caratteri. Inserire il disco System Master nell'unità disco e battere:

LOAD MAKE TEXT

Dovrebbe risultare un listato del programma (LIST) come quello che segue:

```
  5  REM MAKE TEXT
 1Ø  DIM A$(1ØØ):I=Ø
 2Ø D$=CHR$(4): REM CTRL-D
 3Ø  HOME : TEXT
 4Ø  PRINT "YOU GET TO TYPE ONE ST
     RING AT A TIME"
 5Ø  PRINT "A STRING MAY HAVE UP T
     0 239 CHARACTERS."
 6Ø  PRINT "THIS PROGRAM LETS YOU
     WRITE TEXT FILES.
 7Ø  PRINT :I=i+1
 8Ø  PRINT "(PRESS THE RETURN KEY
     TO QUIT.)
 9Ø  PRINT "TYPE STRING #";I;": ";
1ØØ  INPUT "";A$(I)
11Ø  IF A$(I) < > "" GOTO 6Ø
12Ø  PRINT
13Ø  INPUT "WHAT FILE NAME?";N$
14Ø  PRINT D$; "OPEN";N$
15Ø  PRINT D$"DELETE";N$
16Ø  PRINT D$;"OPEN";N$
17Ø  PRINT D-;"WRITE";N$
18Ø  PRINT I-1
19Ø  FOR J=1 TO I-1
2ØØ   PRINT A$(J)
21Ø  NEXT J
22Ø  PRINT D$; "CLOSE";N$
```

Una volta che il programma è caricato (LOAD), memorizzarlo (SAVE) su un disco che non sia protetto in scrittura. (Questa fase è necessaria in quanto questo programma, come il programma ANIMALS discusso nel Capitolo 4 crea un nuovo file).

MAKE TEXT è ancora nella memoria dell'Apple? E nell'Unità disco c'è ancora un disco non protetto da scrittura? In questo caso, battere

MON C, I, O

per poter vedere i comandi inviati da e verso il disco. Battere RUN per veder comparire il seguente messaggio:

YOU GET TO TYPE ONE STRING AT A TIME

A STRING MAY HAVE UP TO 239 CHARACTERS

THIS PROGRAM LETS YOU WRITE TEXT FILES

(PRESS THE RETURN KEY TO QUIT.)
TYPE STRING #1:

Battere un numero di stringhe a piacere (se ne possono impostare fino a 1ØØ).

ATTENZIONE: Il programma usa INPUT, per cui non bisogna battere le virgole o i due punti nelle stringhe. Quando si desidera interrompere, basta premere il tasto **RETURN** invece di battere una stringa. Il programma chiede

WHAT FILE NAME?

Scegliere un nome per il file di testo, premere il tasto **RETURN** e come le stringhe vengono inviate al disco, queste appaiono sul video. Innanzittutto compaiono i comandi del disco

OPEN f
DELETE f
OPEN f
WRITE f

(dove la f è sostituita dal nome del file scelto). Questi saranno seguiti da un numero – il numero delle stringhe impostato nel file.
(Questo numero sarà usato dal programma discusso nella parte che segue che richiama il file). Successivamente compariranno le stringhe. Infine, si vedrà comparire il messaggio:

CLOSE f

Qui c'è un'esecuzione **(RUN)** campione del programma MAKE TEXT:

```
THIS PROGRAM LETS YOU WRITE TEXT FILES.

YOU GET TO TYPE ONE STRING AT A TIME.

A STRING MAY HAVE UP TO 239 CHARACTERS.

(TO QUIT, PRESS RETURN KEY FIRST)
TYPE STRING #1: HERE'S STRING 1

(TO QUIT, PRESS RETURN KEY FIRST)
TYPE STRING #2: and my second string

(TO QUIT, PRESS RETURN KEY FIRST)
TYPE STRING #3: ON WE GO

(TO QUIT, PRESS RETURN KEY FIRST)
TYPE STRING #4: enough already!

(TO QUIT, PRESS RETURN KEY FIRST)
TYPE STRING #5:

WHAT FILE NAME? TEST
OPEN TEST
DELETE TEST
OPEN TEST
WRITE TEST
4
HERE'S STRING 1
AND MY SECOND STRING
ON WE GO
ENOUGH ALREADY!
CLOSE TEST
```



Se si apre (OPEN) un file di testo che già esiste e quindi si scrive su di esso (WRITE) (senza cancellare – DELETE il file e riaprirlo – OPEN) si sovrascrive quanto meno una porzione di file. A meno che non si sostituiscano altrettanti caratteri che già esistevano nel vecchio file, i contenuti del nuovo file saranno un misto dei dati scritti (PRINT) sul file nelle due occasioni.

Dapprima compariranno i nuovi caratteri stampati (PRINT) sul file, quindi seguiranno tutte le porzioni del vecchio file che non sono state sovrascritte. Per cancellare tutti i caratteri dal vecchio file, aprire (OPEN) e cancellare (DELETE) il vecchio file prima di aprirlo (OPEN) di nuovo. (Nel programma MAKE TEXT, le righe 14Ø e 15Ø si occupano della "pulizia" di qualsiasi precedente file di testo con lo stesso nome). Per evitare che i programmi sovrascrivano un file, bloccare (LOCK) il file stesso.

Lettura (read) di files sequenziali

Il comando DOS READ consente di richiamare un file di testo. Una volta che viene eseguito un comando READ, quelsiasi successiva istruzione INPUT (o GET in Applesoft) fa riferimento al file specificato anzichè alla programma listato all'inizio del paragrafo precedente. READ, come WRITE, deve essere preceduto dall'apertura (OPEN) del file da usare. Il file deve essere naturalmente chiuso (CLOSE).

```
5   REM RETRIEVE SAMPLE
1Ø  D$=CHR$ (4): REM CHR$(4)
      IS CTRL-D
2Ø   PRINT D$; ''OPEN SAMPLE''
3Ø   PRINT D$; ''OPEN SAMPLE'
4Ø   INPUT A$;, B$,C$
5Ø   FOR I=1 TO 1Ø
6Ø   INPUT W
7Ø   NEXT I
8Ø   PRINT D$; ''CLOSE SAMPLE''
```

Un OPEN deve precedere un READ, e un INPUT (o, in Applesoft, un GET) deve seguire un READ. OPEN e READ devono fare riferimento allo stesso nome del file. Se si esegue **(RUN)** il programma con MON C, I, O in atto, si vede comparire quanto segue:

```
OPEN SAMPLE
READ SAMPLE
?HI HO
??HI HO
??OFF TO THE DISK WE GO
?1
?2
?3
?4
?6
?7
?8
?9
?10
CLOSE SAMPLE
```

Il programma è stato scritto esplicitamente tenendo presente il file SAMPLE: esso presume che il file di testo contenga tre stringhe (A$, B$, C$ nella riga 4Ø) e 1Ø interi (W nella riga 6Ø). Vengono battuti due punti interrogativi quando B$ e C$ sono introdotti (INPUT) in quanto i **RETURN** separavano lẹ risposte multiple INPUT.

Un comando READ viene cancellato dall'uso di qualsiasi comando DOS in un'istruzione PRINT. Il comando DOS "vuoto" (e cioè soltanto **CTRL-D**) ha lo stesso effetto. Anche l'uso dei comandi PR# o IN# cancella un READ.

La sintassi per il comando READ è la stessa di quella per WRITE:

READ f

(Nota: READ viene anche usato con i files ad accesso casuale, vedere Capitolo 8)

Esempio:

READ LETTER
READ CAREFULLY



L'interruzione di un READ in Applesoft con l'uso di **CTRL-C** genera una stringa di REENTER. Per evitare ciò, premere il tasto **RESET** per interrompere il programma.

Sul disco System Master c'è un programma Applesoft che richiama files di testo creati dal programma MAKE TEXT. Disporre il disco System Master nell'unità e scrivere:

LOAD RETRIEVE TEXT

quindi memorizzare (SAVE) il programma sullo stesso mini floppy usato per MAKE TEXT. (Il programma è realmente analogo a MAKE TEXT ed è semplicemente più comodo averli sullo stesso disco).

Un listato (LIST) del programma dovrebbe comparire come segue:

```
 10 D$=CHR$ (4): REM CTRL D
 12  PRINT ''THIS PROGRAM RETRIEVES
     TEXT FILES''
 14  PRINT ''CREATED BY THE 'CREATE
     TEXT' PROGRAM.''
 16  PRINT ''MON C, I, O IS IN EFFECT
     ''
 18  PRINT
 20  INPUT ''NAME OF TEXT FILE?'';Z
     $
 22  PRINT D$; ''MON C, I, O''
 24  PRINT
 30  PRINT D$; ''OPEN'';Z$
 40  PRINT D$; ''READ'';Z$
 50  INPUT I
 55  DIM A$(I)
 60  FOR J=1 TO I
 70   INPUT A$(J)
 80  NEXT J
 90  PRINT D$; ''CLOSE'';Z$
100  PRINT D$; ''NOMON C, I, O''
```

Battere ora

```
RUN
```

e si vedrà comparire il messaggio

```
THIS PROGRAM RETRIEVES TEXT FILES
CREATED BY THE 'MAKE TEXT' PROGRAM
MON C, I, O IS IN EFFECT

NAME OF TEXT FILE?
```

Battere il nome del file di testo creato usando il programma MAKE TEXT, premere il tasto **RETURN,** per poter terminare.

Ecco ciò che si può osservare se il file TEST usato come esempio al termine dell'ultimo paragrafo viene richiamato usando il programma RETRIEVE TEXT:

```
THIS PROGRAM RETRIEVES TEXT FILES
CREATED BY THE 'MAKE TEXT' PROGRAM.
MON C, I, O IS IN EFFECT.

NAME OF TEXT FILE? TEST

OPEN TEST
READ TEST
?4
?HERE'S STRING 1
?AND MY SECOND STRING
?ON WE GO
?ENOUGH ALREADY!
CLOSE TEST
NOMON C, I, O
```

Ancora sui files sequenziali: append e position

I comandi APPEND e POSITION, consentono rispettivamente di aggiungere del testo al termine di un file di testo sequenziale e di accedere alle informazioni da qualsiasi campo specificato all'interno di un file di testo.

APPEND consente di aggiungere dati alla fine di un file di testo sequenziale. Ciò è particolarmente utile se si desidera estendere l'informazione in un file di testo sequenziale, come nel programma ANIMALS discusso nel Capitolo 4. Il comando OPEN, come si ricorderà, definisce sempre il puntatore di posizione nel file al byte Ø, il primo carattere nel file.

Il comando APPEND esegue un OPEN sul file che già esiste, quindi, definisce il puntatore della posizione programma costruisce un file denominato TESTER CHE CONTIENE le due stringhe "TEST Ø" e "TEST 1":

```
 5  REM MAKE TESTER
1Ø d-=CHR$ (4): REM CTRL-D
2Ø  PRINT D$; "OPEN TESTER"
3Ø  PRINT D$; "DELETE TESTER"
4Ø  PRINT D$; "OPEN TESTER"
5Ø  PRINT D$; "WRITE TESTER"
5Ø  PRINT D$; "WRITE TESTER"
6Ø  PRINT "TEST Ø"
7Ø  PRINT "TEST Ø"
7Ø  PRINT "TEST 1"
8Ø  PRINT D$; "CLOSE TESTER"
```

Il seguente programma aggiunge (APPEND) le stringhe "TEST 2" "TEST 3" e "TEST 4" al file TESTER:

```
 5  REM APPEND TESTER
1Ø D$=CHR$ (4): REM CTRL-D
2Ø  PRINT D$; "APPEND TESTER"
3Ø  PRINT D$; "WRITE TESTER"
4Ø  PRINT "TEST 2"
5Ø  PRINT "TEST 3"
6Ø  print "TEST 4"
7Ø  PRINT D$; "CLOSE TESTER"
```

Il seguente programma presenta il file TESTER:

```
1Ø  REM RETRIEVE TESTER
2Ø D$=CHR$ (4): REM CTRL-D
3Ø  PRINT D$; "OPEN TESTER"
4Ø  PRINT D$; "READ TESTER"
5Ø  FOR I=1 TO 5
6Ø  INPUT A$
7Ø  NEXT I
8Ø  PRINT D$; "CLOSE TESTER"
```

APPEND deve essere seguito da WRITE (un tentativo di leggere (READ) fa comparire il messaggio END OF DATA). La sintassi per il comando APPEND è senza dubbio familiare;

```
APPEND f [,Ss] [,Dd] [,Vv]
```

APPEND quantunque sia usato soltanto per scrivere (WRITE) in un file di testo, non fa comparire il messaggio

```
FILE LOCKED
```

se il file è bloccato. Questo messaggio compare soltanto se si tenta di scrivere (WRITE) effettivamente sul file. Il comando DOS POSITION consente di scrivere (WRITE) o leggere (READ) informazioni cominciando in qualsiasi dato campo di un file di testo sequenziale. La sintassi per il comando POSITION è

POSITION f [,Rp]

dove Rp è la posizione relativa del campo. Questo comando specifica che la posizione del DOS del puntatore di posizione nel file sarà spostata **in avanti** (soltanto) fino al campo p-esimo oltre la posizione corrente del puntatore. Se p=Ø, il seguente READ o WRITE inizia nel campo corrente. Se p=1, il seguente READ o WRITE salta il campo corrente e inizia nel campo **successivo**.

Se p=2, il READ o WRITE seguente salta due campi compreso il campo corrente, prima di iniziare a leggere (READ) o a scrivere (WRITE). E cosi via. Se il file non contiene qualsiasi campo corrispondente alla posizione di campo relativa specificata dal comando POSITION, compare il messaggio

END OF DATA

e l'esecuzione del programma si interrompe.

POSITION con il parametro Rp specifica una posizione di campo **relativa**, a p campi oltre il campo corrente. POSITION deve fare riferimento ad un file che è già stato aperto (OPEN). OPEN definisce automaticamente PUNTATORE DELLA POSIZIONE NEL FILE RIPORTANDOLO ALL'INIZIO DEL PRIMO CAMPO. Cosi, se POSITION viene usato immediatamente dopo OPEN, anche la posizione relativa del campo corrisponde a quella "**effettiva**" o assoluta. In nessun altro caso ciò è vero.

Come qualsiasi altro comando DOS, POSITION cancella un READ o WRITE. Pertanto, position deve essere usato **prima** dei corrispondenti READ o WRITE.

POSITION analizza effettivamente il contenuto del file, byte per byte, alla ricerca del Rp- esimo carattere **RETURN.** Se durante questo processo incontra un byte "vuoto" (valore Ø), compare immediatamente il messaggio

END OF DATA

Ecco un programma che usa POSITION per richiamare vari campi dal file TESTER, creato precedentemente dai programmi MAKE TESTER e APPEND TESTER:

```
10  REM    POSITION TESTER
20  D$ =   CHR$ (4): REM CTRL-D
30  PRINT D$; "OPEN TESTER"
40  PRINT D$; "POSITION TESTER,R2"
50  PRINT D$; "READ TESTER"
60  INPUT A$
70  PRINT D$; "POSITION TESTER, R1"
80  PRINT D$; "READ TESTER"
90  INPUT B-
100 PRINT D$; "OPEN TESTER"
110 PRINT D$; "POSITION TESTER, R3
  "
```

```
120 PRINT D$; "READ TESTER"
130 INPUT C-
140 INPUT E$
150 PRINT D$; "CLOSE TESTER"
```

Se si esegue **(RUN)** con MON C, I, O in atto si vede comparire:

```
OPEN TESTER
POSITION TESTER, R2
READ TESTER
?TEST 2
POSITION TESTER, R1
READ TESTER
?TEST 4
OPEN TESTER
POSITION TESTER, R3
READ TESTER
?TEST 3
?TEST 4
CLOSE TESTER
```

Sorpresi del risultato? Ricordare che il campo corrente è il numero di posizione relativo Ø del campo. Ricordare inoltre che ciascun INPUT provoca la lettura di un campo (READ) in Apple e fa avanzare il puntatore di posizione nel file all'inizio del campo successivo.

Qualcosa in più

Nota: Il paragrafo che segue non è per i i principianti, e i files sequenziali possono essere usati perfettamente anche senza la conoscenza dei parametri qui discussi:

I comandi DOS, WRITE e READ possono essere usati con un parametro BYTE per scrivere (WRITE) o leggere (READ) informazioni a partie da qualsiasi punto in un file di testo -- purchè si conosca dove è quel punto. Il trucco comporta la conoscenza esatta del byte nel file al quale si desidera iniziare (ciascun byte contiene il codice ASCII di un carattere). Per fare ciò, si deve conoscere esattamente come sono state caricate le informazioni nel file. Quando si cerca il punto dal quale cominciare, si devono contare tutti i **RETURN**, le virgole, gli spazi e gli altri caratteri nel file. Il problema è anche più difficile per WRITE, in quanto si deve anche sapere dove terminare.

Il parametro B è una posizione effettiva o assoluta nel file, a meno che R non sia specificato. Se viene specificato R, il parametro B è la posizione effettiva all'interno del campo specificato.

Il comando

```
WRITE THISMONTH, B27
```

definisce il puntatore di posizione nel file al 28° byte del file denominato THISMONTH (il primo byte è il numero Ø). I caractteri inviati al disco dal successivo comando PRINT sostituiscono un numero uguale di caratteri che già esistevano nel file, iniziando il carattere nel 28° byte.



Questa sovrascrittura non è limitata al campo corrente. Se si batte (PRINT) un numero minore di caratteri di quelli rimanenti nel campo corrente, si creano **due** nuovi campi: il campo appena battuto (PRINT), seguito dall'estremità di coda del campo sovrascritto. Se si batte (PRINT) un numero maggiore di caratteri di quelli rimanenti nel campo corrente, si ovrascrivono alcuni dei caratteri all'inizio del campo **successivo**: il campo corrente sarà più lungo, e il successivo campo più corto di prima.

È anche possibile scrivere (WRITE) bytes che sono **oltre** l'ultimo byte di un file di testo sequenziale esistente. Un tentativo di leggere (READ) i bytes intermedi non scritti provoca la comparsa del messaggio END OF DATA e il programma si interrompe. Vedere la discussione DI READ con il parametro B per le informazioni sull'accesso ai campi dei files di testo sequenziali vicini l'uno all'altro.

La sintassi per questo comando è

```
WRITE f [,Bb]
```

dove il parametro B specifica il byte del file al quale i caractteri inviati dal successivo comando PRINT INIZIERANNO A SOSTITUIRE I CARATTERI DEL FILE. Il valore presunto di B è Ø, il primo byte in un file. Il byte b è una posizione **effettiva**, o assoluta all'interno del file. Il parametro b può specificare una posizione prima o dopo il puntatore della posizione corrente nel file.
(Nota: Questo comando viene anche usato con i files ad accesso casuale. Vedere Capitolo 8).

Analogamente, il comando

```
READ LASTMONTH, B32
```

definisce il puntatore della posizione nel file al 33esimo byte del file denominato LASTMONTH (anche qui, il primo byte è il numero Ø). Un successivo comando INPUT fa si che tutti i caratteri nel successivo campo (e cioè fino al successivo carattere RETURN) a partire dal carattere il cui codice ASCII è caricato nel 33esimo bytes del file, vengano letti (READ) in Apple, Se il 33 esimo byte non contiene il primo carattere di un campo, vengono letti soltanto i rimanenti caratteri di quel campo.

La sintassi per questo comando è

```
READ f [,Bb]
```

dove il parametro B specifica il byte del file in cui i successivi comandi INPUT o GET inizieranno a leggere i caratteri. Il valore presuntodib è Ø, il primobyte in un file. Il byte b è una posizione **effettiva** o assoluta all'interno del file. Il parametro b può specificare una posizione prima o dopo il puntatore corrente di positione nel file. (Nota: Questo comando è anche usato con i files ad accesso casuale, vedere Capitolo 8).

Il seguente programma definisce il puntatore di posizione nel file al byte 14 (il 15° byte) nel file TESTER creato precedentemente dal programma MAKE TESTER. Quindi scrive (WRITE) la stringa "APPLE COMPUTER". Notare la sequenza familiare: OPEN, quindi WRITE e PRINT e infine CLOSE.

```
 5  REM BYTE WRITER
10  D$=CHR$ (4): REM CTRL-D
20  PRINT D$; "OPEN TESTER"
30  PRINT D$; "WRITE TESTER, B14"
40  PRINT "APPLE COMPUTER"
50  PRINT D$; "CLOSE RESTER"
```

Con MON C, I, O in atto, eseguire **(RUN)** RETRIEVE TESTER per vedere come il precedente programma ha cambiato il file TESTER. Come si può vedere, il campo contenente APPLE COMPUTER ha completamente sostituito i campi che contenevano TEST 3 e TEST 4, nonchè il primo carattere del campo che conteneva TEST 5. Dato che ci sono ora solo 4 campi in tutto, il messaggio

END OF DATA

è stato presentato dopo il 5° comando INPUT.

Il seguente programma definisce un puntatore al byte 18 nel file TESTER appena modificato dal precedente programma. Quindi questo programma legge (READ) al successivo **RETURN** nel file. Di nuovo, il formato familiare: OPEN seguito da READ e successivamente vengono le istruzioni INPUT (oppure in Applesoft, possono essere usati i GET) e infine il file è chiuso (CLOSE).

```
 5  REM BYTE RADER
10  D-=CHR$ (4): REM CTRL-D
20  PRINT D$; "OPEN TESTER"
30  PRINT D$; "READ TESTER, B18"
40  INPUT A$
50  PRINT D$; "CLOSE TESTER"
```

Cercare di prevedere cosa si vedrà sul video prima di eseguire **(RUN)** questo programma.

CAPITOLO 7

Apple "Automatico"

Per comprendere meglio il contenuto di questo Capitolo, si consiglia di leggere dapprima il Capitolo 6, relativo ai files di testo sequenziali.

Controllo di apple attraverso un file di testo : EXEC

Il comando DOS EXEC è simile a **RUN,** salvo che il file su disco usato da un comando EXEC è un file di testo che contiene comandi o righe di programma, comprendenti istruzioni BASIC, come se fossero battuti alla tastiera.

Per iniziare una dimostrazione di alcune capacità di comandi EXEC, caricare

LOAD EXEC DEMO

dal disco System Master e quindi memorizzare (SAVE) su un disco non protetto da scrittura. Lasciare il disco non protetto in scrittura nell'unità disco, dato che il programma scrive (WRITE) un file di testo.

Successivamente eseguire **(RUN)** il programma. Si dovrebbe vedere comparire il messaggio

```
                    ''EXEC DEMO''
THIS PROGRAM CREATES A SEQUENTIAL TEXT FILE NAMED ''DO/ER''
CONTAINING SEVERAL STRINGS, EACH A LEGAL APPLE II COMMAND.
WHEN YOU TYPE
EXEC DO'ER
THEN THE COMMANDS IN FILE DO'ER TAKE CONTROL OF YOUR
COMPUTER. EACH COMMAND WILL BE EXECUTED JUST AS IF IT HAD BEEN
TYPED AT THE KEYBOARD. THE DOS MANUAL DESCRIBES THE PROGRAM
IN MORE DETAIL.
                 ''HAPPY EXECUTING''
PRESS THE SPACE BAR TO MAKE THIS PROGRAM CREATE THE FILE DO'ER.
IF YOU WISH TO STOP THIS PROGRAM NOW YOU MAY PRESS THE ESC KEY.
```

Premere la barra di spazio di Apple e dopo una breve pausa si dovrebbe accendere la spia IN USE dell'unità disco quindi il programma scrive il file DO'ER sul mini floppy. Battere ora

EXEC DO'ER

e premere il tasto **RETURN.** Apple inizia un "a solo" basato sul testo nel file DO'ER.

Ecco un breve riassunto delle principali cose che DO'ER fa :
– innanzitutto DO'ER emette un comando MON C, I, O cosìcchè è possibile vedere cosa succede.
– Secondo, viene scritto un programma di 3 righe che viene memorizzato sul disco sotto il nome NEW PROGRAM!! Il programma viene quindi listato (LIST).
– Viene ora eseguito un loop FOR-NEXT per guadagnare tempo, in modo da avere la possibilità di osservare lo schermo prima che l'attività continui.
– Successivamente, DO'ER usa il comando INT per impostare l'Integer BASIC, carica (LOAD) il programma COLOR DEMOS e lo elenca (LIST). A questo punto, DO'ER usa CALL-155 per inserire il Monitor ed eseguire alcune istruzioni in linguaggio macchina prima di usare il comando FP per inserire Applesoft.
– Da Applesoft, viene eseguito un comando MON C, I, O quindi viene eseguito (**RUN**) NEW PROGRAM!! modificato, listato (LIST), (di nuovo un loop FOR consente di dare uno sguardo al video) e memorizzato (SAVE) usando il nome EVEN MORE RECENT PROGRAM!!

– Infine, il programma NEW PROGRAM!! viene cancellato (DELETE) e viene presentato il CATALOG (compresa la nuova aggiunta EVEN MORE RECENT PROGRAM!!).
E non c'è da mettere un dito sulla tastiera (a meno che il CATALOG non abbia più di 18 entrate, nel qual caso occorre premere la barra di spazio per vedere il resto delle entrate CATALOG).

Creazione di un file EXEC

Segue ora un esempio passo passo per creare un file EXEC denominato DOIT, che contiene i seguenti comandi

```
LIST 2Ø, 5Ø
RUN AWAY
CATALOG
```

Creare dapprima e memorizzare (SAVE) un programma Applesoft denominato AWAY da usare nella suddetta dimostrazione:

```
 5  REM AWAY
1Ø  PRINT "AWAY AWAY WITH RUM BY GUM"
```

scrivere quindi il programma seguente denominato MAKE EXEC che crea un file di testo DOIT. Quando successivamente si esegue EXEC DOIT, i comandi che il programma MAKE EXEC ha scritto (PRINT) nel file di testo DOIT diranno ad Apple di eseguire (**RUN**) il programma AWAY. Notare che i comandi stampati (PRINT) nel file DOIT, per la successiva esecuzione (EXEC) non sono preceduti da un CTRL-D.

```
 5  REM MAKE EXEC
1Ø  D$ = CHR$ (4) : REM CHR$(4) IS CTRL-D
2Ø  PRINT (D$; "OPEN DOIT"
3Ø  PRINT D$; "WRITE DOIT"
4Ø  PRINT "LIST 2Ø, 5Ø"
5Ø  PRINT "RUN AWAY"
6Ø  PRINT "CATALOG"
7Ø  PRINT D$; "CLOSE DOIT"
```

Dopo aver memorizzato (SAVE) sia MAKE EXEC che AWAY su un disco, battere il comando

```
RUN MAKE EXEC
```

per creare un file di testo sequenziale DOIT.
Battere il comando

```
EXEC DOIT
```

per far sì che i comandi nel file DOIT vengano eseguiti uno per uno, esattamente come se fossero battuti dalla tastiera. Anche qui notare che i comandi che vengono ora eseguiti (EXEC) non erano preceduti da un **CTRL-D** nel programma MAKE EXEC. Le prime righe da 2Ø a 5Ø del programma correntemente in memoria (probabilmente il programma MAKE EXEC) sono listate (LIST). Quindi, viene eseguito (**RUN**) il programma denominato "AWAY" e infine viene presentato il CATALOG del mini floppy.

Cattura di programmi in un file di testo

Qui c'è un esempio molto più utile dell'uso del comando EXEC: esso consente di catturare listati di programma come files di testo. Tale programma può essere usato per:

* Tradurre programmi Integer BASIC in Applesoft
* Numerarte parti di programmi ed eseguirle (EXEC) ovunque in un altro programma
* Inserire subroutines preferite in programmi da un file di subroutines sul mini floppy eseguendo (EXEC) il file
* "Aggiungere" un programma ad un altro
* "Riparare" programmi che sono diventati parzialmente illeggibili (è possibile catturare la parte buona in un file di testo, rilanciarla quindi riportare (EXEC) la parte di programma in memoria.

I numeri di riga 227∅ et 513∅ che seguono il comando LIST nella riga 6 del programma CAPTURE, devono essere sostituiti ai numeri di riga del programma in memoria che si desidera catturare. Il nome del file di testo sequenziale contenente il listato è LISTING.

```
1  REM CAPTURE
2 D$=CHR$ (4): REM CTRL-D
3  PRINT D$; ''OPEN LISTING''
4  PRINT D$; ''WRITE LISTING''
5  POKE 33, 3∅
6  LIST 227∅,513∅
7  PRINT D$; ''CLOSE LISTING''
8  TEXT: END
```

I numeri di riga di questo programma sono stati definiti l'uno molto vicino all'altro, cosìcchè è possibile aggiungere queste righe a un programma già in memoria o altrove all'interno del programma in cui ci sono 8 numeri di riga liberi. Si possono anche facilmente inserire tutte le righe CAPTURE al di sopra della riga di numero più elevato nel programma. CAPTURE crea un file di testo contenente i comandi preceduti dai numeri di riga. Quando si esegue (EXEC) quel file di testo, i comandi coi numeri alti non vengono eseguiti. Per contro, come se si fossero battute queste righe dalla tastiera, le righe vengono caricate come programma nella memoria di Apple. Una volta catturato in un file di testo, un programma può essere modificato e quindi eseguito (EXEC) nella memoria di Apple. A differenza di LOAD o **RUN**, EXEC non cancella un programma che è già in memoria. Usando CAPTURE è possibile catturare un programma in un file di testo in un linguaggio, quindi eseguirlo (EXEC) in un altro linguaggio (naturalmente il programma potrebbe non girare senza alcune modifiche — c'è una sintassi alquanto diversa per Integer BASIC e Applesoft).

È anche possibile usare EXEC in questo modo per aggiungere nuove righe ad un programma esistente in memoria. In effetti, si può memorizzare un listato di CAPTURE in un file di testo denominato LIST SAVER, ad esempio, e quindi EXEC LIST SAVER in qualsiasi momento si desidera per aggiungere il programma CAPTURE ad un programma già in memoria.

Conversione in basic di routines in linguaggio di macchina

Ecco un altro utile programma che prende una routine in linguaggio di macchina e la converte in una porzione di programma BASIC che inserisce (POKE) la routine in linguaggio macchina in memoria. La porzione del programma può essere usata come parte di un programma Applesoft o Integer BASIC, per inserire la routine in linguaggio macchina ogni volta che viene eseguito il programma BASIC.

```
  5 REM CODE-POKES WRITER
 1Ø D$="": REM CTRL-D
 15  PRINT D$; "OPEN CODE-POKES"
 2Ø  PRINT D$; "DELETE CODE-POKES"
 25  PRINT D$; "OPEN CODE-POKES"
 3Ø  PRINT D$; "WRITE CODE-POKES"
 4Ø LINENUMBER=7ØØØ
 5Ø  FOR PLACE=768 TO 783
 6Ø COUNTER=COUNTER +1
 7Ø  IF COUNTER=1Ø THEN COUNTER=
     1
 8Ø  IF COUNTER < > 1 THEN 12Ø
 9Ø  PRINT
1ØØ  PRINT LINENUMBER;
11Ø  LINENUMBER=LINENUMBER + 1
12Ø  PRINT"POKE"; PLACE;",";  PEEK
     (PLACE);":";
13Ø  NEXT PLACE
135  PRINT
14Ø  PRINT D$: "CLOSE CODE-POKES"
15Ø  END
```

Quando si usa questo programma, il numero della riga 4Ø deve essere cambiato per contenere il numero di riga del programma BASIC in cui deve iniziare l'inserimento della porzione del programma (POKE). Il loop FOR nella riga 5Ø deve contenere le posizioni di memoria decimali di inizio e di fine della routine in linguaggio di macchina che si desidera convertire.

Una volta battuto il programma, la sua esecuzione (**RUN**) crea i files di testo CODE-POKES.
Usare ora il comando

```
EXEC CODE-POKES
```

per disporre la porzione di programma (POKE) in linguaggio di macchina in qualsiasi altro programma, iniziando dal numero di riga precedentemente specificato. Il programma CODE-POKES WRITER funziona sia con Applesoft che con Integer BASIC.

MAXFILES e i programmi integer BASIC

Si deve usare un file EXEC se si desidera aumentare MAXFILES dall'interno di un programma Integer BASIC senza cancellare i programmi esistenti. Ecco come usare le procedure sopra descritte per creare un file EXEC che sarà denominato FILE.EX.

FILE.EX deve contenere i sequenti comandi per accogliere 5 files su un sistema:

```
MAXFILES 5
LOAD PROGRAM
DEL 1Ø, 2Ø
RUN
```

Le prime righe del programma sarebbero come segue; notare ciò che compare quando viene creato **CTRL-D** tenendo abbassato il tasto contrassegnato **CTRL** mentre si batte la lettera D.

```
1Ø PRINT "CTRL-D EXEC FILE.EX"
2Ø END
3Ø MAIN PROGRAM
```

Un esempio di esecuzione automatica

La sintassi abituale per il comando EXEC è

```
EXEC f
```

dove f è il nome di un file di testo sequenziale contenente comandi BASIC o righe di programma. Esempi di questo uso compaiono nei primi paragrafi di questo Capitolo. EXEC con questa sintassi fa si che il campo del file f venga letto in Apple come se fosse battuto sulla tastiera. Quando il carattere RETURN del primo campo è "battuto" Apple tenta di eseguire i contenuti del campo come un comando BASIC, oppure di impostare i contenuti del campo come un comando BASIC, oppure di impostare i contenuti del campo come una riga di programma BASIC. Il tipo di BASIC (Integer o Applesoft) non viene cambiato da EXEC, a meno che il file non contenga un comando FP oppure INT. Quando l'esecuzione è terminata sul primo campo, il secondo campo del file f viene letto in Apple e trattato analogamente. L'azione si interrompe quando è stato letto l'ultimo campo del file f.

Il comand EXEC non può essere interrotto da **CTRL-C.**

In un qualsiasi momento può essere aperto un solo file EXEC. Durante l'esecuzione (EXEC) di un file e quando uno dei comandi così eseguití è un altro comando EXEC, il primo file EXEC viene immediatamente chiuso. Successivamente, viene eseguito il secondo comando EXEC.

Se viene eseguito un file (EXEC) che contiene un comando per eseguire **(RUN)** qualsiasi programma, EXEC attende pazientemente fino a che il programma non termina. Quindi, viene eseguito il successivo comando del file EXEC.



Per contro, se un programma è in esecuzione **(RUN)** mentre è aperto un file EXEC (OPEN) qualsiasi istruzione di input nel programma prende come risposta il successivo campo nel file che viene eseguito (EXEC), ignorando la

tastiera. Peggio ancora, se quella risposta è un comando DOS ad esecuzione immediate, il comando viene **eseguito** prima che il programma continui. I risultati possono essere molto strani.



Se si interrompe l'esecuzione **(RUN)** di un programma Applesoft battendo **CTRL-C** mentre è aperto un file EXEC (OPEN) il resto di quel file EXEC non viene solitamente eseguito.

Se qualsiasi campo di un file EXEC non può essere interpretato come un comando o riga di programma BASIC valido, viene generato il messaggio

SYNTAX ERROR

e in Apple viene letto il successivo campo. Pertanto, è possibile eseguire (EXEC) **qualsiasi** file di testo, contenga esso o meno istruzioni BASIC (assicurarsi prima di aver memorizzato – SAVE – il programma in memoria). Nel modo MON C, I, O, ciò può rappresentare uno strumento grezzo ma utile per esaminare rapidamente i contenuti di un file di testo.

Il comando EXEC può anche essere usato con il parametro di posizione relativa di campo, in un modo che è un po' diverso dall'uso di POSITION di quel parametro. La sintassi in questo caso è.

EXEC f [,Rp]

dove Rp specifica che il file f deve essere eseguito (EXEC) iniziando nel campo p-esimo del file f. Dato che EXEC definisce sempre il puntatore della posizione nel file, al primo carattere del file, cosi il parametro Rp indica sempre il campo p-esimo rispetto all'**inizio** del file e pertanto p corrisponde sempre al campo **effettivo** o assoluto del file. RØ indica che EXEC inizia con il primo campo del file. R1 indica che EXEC inizia col secondo campo, ecc.



Notare che ciò è diverso dall'uso di POSITION del parametro R, dove R3 è soltanto un campo **relativo** e può indicare campi di file effettivi diversi in tempi diversi in un programma.



EXEC MYFILE, Rf

genera un messaggio

END OF DATA

se il parametro R specifica il **secondo** campo oltre il termine del file (Se è specificato il **primo** campo oltre il termine del file non succede nulla).

CAPITOLO 8

Uso dei files ad accesso casuale

Per una maggior comprensione delle informazioni presentate in questo Capitolo, si consiglia di leggere prima il Capitolo 6 sui files sequenziali.

I files ad accesso casuale : come funzionano

I files ad accesso casuale sono come una raccolta di cellule di uguali dimensioni in un alveare – alcune delle quali possono essere piene, altre vuote. Ciascuna "cellula" è detta **record.** Quando si crea un file ad accesso casuale, si deve specificare la dimensione standard dei records che il file deve contenere.

A differenza dei campi nei files sequenziali, che possono avere pressochè qualsiasi lunghezza, i records in un file ad accesso sequenziale devono essere di una **lunghezza** fissa specificata. La prima volta in cui si scrive (WRITE) su un particolare record in un file, su disco viene accantonato sufficiente spazio per un record completo di lunghezza standard, indipendentemente dal fatto che il record sia o meno effettivamente riempito. Così i files ad accesso casuale non rappresentano necessariamente un uso efficiente dello spazio. Per contro, dato che questi files sono creati in questo modo regolare, si possono facilmente e velocemente richiamare o modificare le informazioni in una parte qualsiasi del file – da qui il nome di file "ad accesso casuale".

I files ad accesso casuale devono essere usati nelle applicazioni che richiedono un accesso veloce alle varie parti del file, oppure dove i singoli elementi di informazione del file devono essere cambiati abbastanza spesso. Ad esempio, un file ad accesso casuale è particolarmente adatto per gestire un elenco di distribuzione.

I files ad accesso casuale sono creati e richiamati in modo analogo a quello usato per i files sequenziali. La differenza principale è che taluni comandi presentano ulteriori parametri. OPEN richiede un parametro L (lunghezza del record), mentre READ e WRITE usano ciascuno un parametro R (numero del record). Verranno ora presentati e discussi alcuni programmi-esempio prima di entrare nei particolari sul modo per creare e richiamare i files ad accesso casuale e sul modo in cui questi parametri funzionano. Ulteriori informazioni tecniche sui files di testo ad accesso casuale si possono trovare nell'Appendice C.

Un record specifico

Come è possibile accedere ad un record specifico in un file ad accesso casuale? La seguente coppia di programmi Applesoft illustra come DOS opera in questo caso. Il primo programma richiede un nome (N$), un numero telefonico (P$) e un codice postale (Z$) e quindi li invia al record 1 di un file denominato MAILER :

```
1Ø REM MAKE MAILER
2Ø D$= CHR$ (4) REM CTRL-D
3Ø  INPUT "NAME :        "; N$
4Ø  INPUT "PHONE :       "; P$
5Ø  INPUT "ZIP CODE;     "; Z$
6Ø  PRINT D$; "OPEN MAILER, L2ØØ"
70  PRINT D$; "WRITE MAILER,R1"
8Ø  PRINT N$ : PRINT P$ : PRINT Z$
9Ø  PRINT D$; "CLOSE MAILER"
```

La riga 2Ø inserisce un **CRTL-D** nella variabile D$, come al solito.
Le righe da 3TTB34B1 Ø a 5Ø richiedono l'informazione da caricare. Non battere virgole o punti e virgole nelle risposte.
La riga 6Ø apre (OPEN) un file denominato MAILER, con records di 2ØØ byte di lunghezza.

La riga 7Ø prepara la registrazione dell'informazione nel record 1.
La riga 8Ø invia N$, P$ e Z$ al disco – e dato che il record 1 era specificato alla riga 7Ø, vengono registrati tutti e tre gli elementi di informazione nel record 1, separati da **RETURN.** La riga 9Ø chiude (CLOSE) il file.

Con MON C, I, O in atto, quando il programma viene eseguito si può osservare:

```
NAME :       AMY DOAKS
PHONE :      (425) 555-1Ø1Ø
ZIP CODE: 95Ø14
OPEN MAILER, L2ØØ
WRITE MAILER, R1
AMY DOAKS
(425) 555-1Ø1Ø
9514
CLOSE MAILER
```

Il record 1 nel file mailer può essere richiamato da questo programma:

```
1Ø REM RETRIEVE MAILER : A
2Ø D$ = CHR$ (4) : REM CTRL-D
3Ø  PRINT D$ "OPEN MAILER, L2ØØ"
4Ø  PRINT D$ "READ MAILER, R1"
5Ø  INPUT N1$, P1$, Z1$
7Ø  PRINT D$; "CLOSE MAILER"
```

Quando eseguito **(RUN)** con MON C, I, O si può osservare quanto segue. Come al solito, la coppia di punti di domanda indica un input con più di una risposta.

```
OPEN MAILER, L2ØØ
READ MAILER, R1
?AMY DOAKS
??(425) 555-1Ø1Ø
??95Ø14
CLOSE MAILER
```

E qui c'è un programma leggermente diverso per richiamare il record 1 di MAILER.

```
1Ø REM RETRIEVE MAILER : B
2Ø D$=" " : REM QUOTES CONTAIN
     CTRL-D
3Ø  PRINT D$; "OPEN MAILER, L2ØØ"
4Ø  PRINT D$; "READ MAILER, R1"
5Ø  INPUT N1$
6Ø  INPUT P1$
7Ø  INPUT Z1$
8Ø  PRINT D$; "CLOSE MAILER"
9Ø  END
```

Records multipli

Il programma che ha creato il file ad accesso casuale MAILER ha scritto su un singolo record nel file, memorizzando tre diversi elementi di informazione separati da **RETURN.** Il successivo programma illustra la scrittura su parecchi records : in particolare, i records da 12 a 15 di un file ad accesso casuale denominato RA-FILE.

```
  5  REM MAKE RA-FILE
 1Ø  D$= CHR$ (4) : REM CTRL-D
 2Ø  PRINT D$; "OPEN RA-FILE"
 3Ø  PRINT D$; "DELETE RA-FILE"
 4Ø  PRINT D$; "OPEN RA-FILE, L3Ø"
 5Ø  FOR I = 12 TO 15
 6Ø  PRINT D$; "WRITE RA-FILE, R"; I
 7Ø  PRINT "NAME ADDRESS"; I
 8Ø  NEXT I
 9Ø  PRINT D$; "WRITE RA-FILE, R13"
1ØØ  PRINT "DOS"
11Ø  PRINT D$; "CLOSE RA-FILE"
```

La riga 1Ø definisce D$ a CTRL-D.
Le righe 2àØ e 3Ø si assicurano che RA-FILE sia un nuovo file.
La riga 4Ø apre (OPEN) il file RA-FILE, i cui records avranno ciascuno 3Ø bytes di lunghezza.
Le righe da 5Ø a 8Ø creano un loop che scrive (WRITE) le informazioni NAME ADDRESS seguite dal numero del record, per i records da 12 a 15. Notare che occorre specificare ciascun record con un nuovo comando WRITE, prima che PRINT mandi i caratteri a quel record.

Le righe 9Ø e 1ØØ cambiano le informazioni nel record 13 come da testo indicato nel comando PRINT della riga 1ØØ. La riga 11Ø chiude (CLOSE) il file ad accesso casuale RA-FILE.

Se è in atto MON C, I, O quando il programma viene eseguito **(RUN)** si vede quanto segue :

```
OPEN RA-FILE
DELETE RA-FILE
OPEN RA-FILE, L3Ø
WRITE RA-FILE, R12
NAME ADDRESS 12
WRITE RA-FILE, R13
NAME ADDRESS 13
WRITE RA-FILE, R14
NAME ADDRESS 14
WRITE RA-FILE, R15
NAME ADDRESS 15
WRITE RA-FILE, R13
DOS
CLOSE RA-FILE
```

In modo analogo, è possibile leggere (READ) informazioni da un record selezionato o da un record di un file di testo. Il successivo programma richiama i records da 12 a 15 del file denominato RA-FILE, cercando, alla riga 6Ø, di trovare quale record o quali records contiene o contengono le lettere "DOS" come primi tre caretterri.

```
5   REM RETRIEVE RA-FILE
10  D$ = CHR$ (4): REM CTRL-D
20  PRINT D$; "OPEN RA-FILE, L30"
30  FOR J=12 TO 15
40  PRINT D$; "READ RA-FILE, R"; J
50  INPUT A$
60  IF LEFT$ (A$, 3)="DOS" THEN
    PRINT "RECORD"; J; " WAS CHA
    NGED."
70  NEXT J
80  PRINT D$; "CLOSE RA-FILE"
```

La riga 10 definisce **a CTRL-D** D$.
La riga 20 apre (OPEN) il file di testo RA-FILE i cui records sono lunghi 30 bytes (lunghezza specificata nel momento in cui il file è stato creato in un precedente programma, non è vero?).
Le righe da 30 a 70 leggono (READ) i records da 12 a 15 del RA-FILE.

Notare che occorre specificare **ciascun** record con un nuovo comando READ, prima che un successivo INPUT legga i caratteri da quel record. Nella riga 50, ciascun record arriva dal disco come una stringa ASCII terminata da un **RETURN.** La riga 60 controlla i tre caratteri più a sinistra della stringa di input A$ dal record r, per vedere se è presente la parole "DOS".
Se lo è, viene battuto il messaggio "RECORD r WAS CHANGED" e la ricerca continua.
La riga 80 chiude (CLOSE il file.

Ecco ciò che si vede quando si esegue **(RUN)** il programma se è in atto MON C, I, O :

```
OPEN RA-FILE, L30
READ RA-FILE, R12
?NAME ADDRESS 12
READ RA-FILE, R13
?DOS
RECORD 13 WAS CHANGED
READ RA-FILE, R14
?NAME ADDRESS 14
READ RA-FILE, R15
?NAME ADDRESS 15
CLOSE RA-FILE
```

Notare che quando il file è stato richiamato sono stati esaminati soltanto i records su cui si era scritto. Se fosse stato richiesto il record 8 nel RA-FILE, si sarebbe ricevuto il messaggio

```
END OF DATA
```

dato che mnessuna informazione era stata scritta su quel record del file.
Analogamente, se si fosse tentato di eseguire l'INPUT di più di un campo da uno qualsiasi dei records esistenti, si sarebbe ottenuto lo stesso messaggio : ciascuno dei records da 12 a 15 contiene soltanto un campo.

Una dimostrazione : il programma con accesso casuale

Infine, ultimo ma non per questo meno importante, il disco System Master contiene un programma denominato RANDOM che usa un file di testo ad accesso casuale per dimostrare un piccolo schema di controllo di inventario. E per piccolo intendiamo veramente piccolo : il programma può gestire al massimo 9 voci. Ciò rende il programma molto semplice. Apple, naturalmente, è un grado di gestire migliaia di voci in un inventario.

Innanzitutto il programma copia sè stesso e le PROM APPLE dei files di testo ad accesso casuale usate per tenere traccia dell'inventario, quindi, esegue automaticamente **(RUN)** il programma. È possibile elencare una o tutte le voci dell'inventario. È possibile, inoltre, cambiare le voci, sia una alla volta che tutte in una sola volta. Ecco come. Ricordarsi di premere il tasto **RETURN** ogniqualvolta si completa una risposta.

1. Dal System Master

   RUN RANDOM

   Dovrebbe comparire il messaggio

   THIS DEMONSTRATION WILL NOT EXECUTE ON
   A WRITE-PROTECTED
   DISKETTE SUCH AS
   YOUR DOS SYSTEM MASTER
   FOR YOUR CONVENIENCE, PROVISIONS HAVE
   BEEN MADE TO COPY THIS PROGRAM AND IT'S
   DATA TO ANOTHER DISKETTE

   DO YOU WISH TO DO THIS NOW? (Y OR N) Y

   Se si batte N (per "no") in risposta al suddetto messaggio ci si ritrova ancora in Applesoft.
2. Premere Y (per "si"). Si noterà il messaggio

   NOW READING DATA

   Seguito dal messagio

   INSERT AN INITIALIZED DISKETTE, THEN
   PRESS THE RETURN KEY TO BEGIN TRANSFER

3. Rimuovere il disco System Master e inserirne nell'unità disco uno non protetto da scrittura, quindi, premere il tasto **RETURN.** Si riesce probabilmente ad intravedere di sfuggita un pezzo del messaggio e quindi il programma inizia l'esecuzione.

   WHEN THE PROGRAM AND DATA HAVE BEEN
   FULLY TRANSFERRED, THE PROGRAM WILL
   BEGIN RUNNING.

4) Si dovrebbe ora vedere quanto segue :

COMMAND	APPLE PROMS NUMBER

LIST

CHANGE	2
EXIT	3
CHOOSE NUMBER (1-3)	1

Premendo 1 si dovrebbe vedere il messaggio seguente:

PART NUMBER 1-9 (Ø=ALL) Ø

5. Premere Ø per ottenere un elenco di tutte le "voci" in questo "sistema di inventario". Sullo schermo comparirà:

PART#	NAME	SIZE	IN STOCK
1	PARALLEL PRINT	256	5ØØ
2	COMMUNICATIONS	256	125Ø
3	(NOT AVAILABLE)	256	Ø
4	(NOT AVAILABLE)	256	Ø
5	DISK BOOT	256	432
6	STATE MACHINE	256	46Ø
7	SERIAL PRINTER1	256	878
8	SERIAL PRINTER2	512	741
9	CENTRONICS	256	129Ø

PRESS THE RETURN KEY TO CONTINUE.

Quando è il momento di tornare all'elenco delle options, premere il tasto **RETURN.**

6. Provare le varie options del programma. La scelta 1 consente di elencare le parti per numero di voce, una alla volta, nonchè tutte in una volta.

   La scelta 2 consente di cambiare uno qualsiasi o tutti i nomi delle voci e le relative descrizioni. Ad esempio, si supponga che la voce 3 debba essere denominata COSMIC GLUE, misura 56 con 1234 pezzi in scorta. Ecco come revisionare l'impostazione della voce 3:
   scegliere l'option 2, CHANGE
   scegliere il numero di voce 3
   Il vecchio nome della voce viene presentato con il cursore all'inizio per consentire di impostare il nuovo nome;
   quando si preme il tasto **RETURN,** il cursore si sposta alla destra e si comporta analogamente per la misura della voce e la sua quantità.
   Per usare il nome o la quantità o la misura correntemente esistenti, basta premere il tasto **RETURN.**

La scelta 3 interrompe il programma.

Scrittura (WRITE) e lettura (READ) di files di testo ad accesso casuale

Quando usato con files di testo di accesso casuale, il comando CLOSE funziona esattamente come con i files sequenziali (vedere "Apertura e chiusura dei files sequenziali" nel Capitolo 6). In ogni caso, la sintassi OPEN ha un ulteriore parametro, il parametro L, che è obbligatorio.

OPEN f, ,Lj [,Ss] [,Ds] [,Vv]

"L" sta per "lunghezza del record"; il numero j indica quanti bytes (caratteri e cifre) devono essere assegnati a ciascun record nel file ad accesso casuale che si sta creando (o se si sta richiamando un file, il numero che era stato assegnato quando il file è stato creato). Se la option L è omessa, a j viene assegnato il valore presunto. Il numero j deve essere nel campo da 1 a 32767.



Quando si apre (OPEN) un file prima della sua lettura (READ) se si specifica un parametro di lunghezza diverso da quello qpecificato al momento dell'apertura, prima della scrittura, DOS usa ciecamente il nuovo parametro di lunghezza per calcolare le posizioni dei records all'interno del file. Occorre, pertanto, mantenere una dettagliata documentazione scritta della struttura e dei contenuti dei files (alcuni programmatori tengono tali informazioni nel record Ø del file). È di aiuto includere sempre il parametro di lunghezza in ciascun nome e file, con nomi tipo :

RANDFILES : L2Ø
STOCKLISTS-L 1ØØ
DIRECTORIES(L5Ø)

Non c'è modo di trovare la lunghezza di un record in un file ad accesso casuale : occorre sempre che queste informazioni facciano parte della documentazione.



I records non devono essere mai più lunghi del numero di bytes specificato dal parametro l : i record possono essere parzialmente sovrascritti o combinati con risultati che generano confusione.
WRITE e READ hanno ciascuno un parametro R, da usarsi quando si creano o si richiamano particolari records nei files ad accesso casuale.

WRITE f [,Rr]
READ f [,Rr]

Esempi : WRITE LEGIBLY, R3
READ FAST, R13

Il parametro Rr (record) è usato per creare (con XRITE) o richiamare (con READ) il record r-esimo del file. Il valore presunto di r è Ø, che specifica il primo record di un file.



L'uso di **CTRL-C** per interrompere un READ in Applesoft provoca la generazione di una stringa di REENTER (rientri); premere in questo caso il tasto **RESET.**

Sotto alcuni aspetti, ciascun record separato in un file di testo ad accesso casuale può essere trattato come un breve file sequenziale. WRITE e READ possono essere usati con un parametro Byte in aggiunta al rispettivo parametro R. Il parametro Byte specifica il byte di inizio del record specificato, per il successivo comando PRINT (dopo WRITE) o INPUT o GET (dopo READ).

WRITE f [,Rr] [,Bb]
READ f [,Rr] [,Bb]

Se specificato, il parametro B (Byte) provoca l'inizio della scrittura (WRITE) o della lettura (READ) al byte b-esimo del record specificato. Il valore presunto di b è Ø, il primo byte del record. Il parametro B può specificare una posizione nel record sia prima che dopo il puntatore corrente della posizione nel file. L'uso del parametro B necessita di una conoscenza dettagliata e approfondita, byte per byte, dei contenuti di ciascun record nel file.

Una volta che READ o WRITE hanno spostato il puntatotre della posizione del file su un particolare record, può anche essere usato POSITION per spostare il puntatore in avanti (soltanto) versoulteriori posizioni relative di campo all'interno del record. Per contro, POSITION controlla sia il modo WRITE che il modo READ (senza cambiare il puntatore della posizione nel file), per cui è necessario un altro comando WRITE o READ (stavolta senza parametro) per reinstaurare quel modo.

I particolari sul modo in cui le informazioni vengono memorizzate sul disco, in generale, e sui files ad accesso casuale in particolare, si possono trovare nell'Appendice C.

CAPITOLO 9

Uso dei files in linguaggio macchina

Files in linguaggio macchina

Il DOS consente di caricare su disco e richiamare da disco le informazioni nella memoria di Apple II. Si sono già visti i comandi DOS, SAVE, LOAD e **RUN** : questi comandi si occupano dei contenuti della memoria programma di Apple, interpretati come comandi nei programmi BASIC.I comandi DOS discussi in questo Capitolo – BSAVE, BLOAD e BRUN – eseguono funzioni analoghe, ma trattano con i contenuti di qualsiasi porzione della memoria di Apple, in forma non interpretata, cioè binaria in formato esadecimale.

Il B prima di ciascuno dei seguenti comandi sta per file Binario; una B precede inoltre il nome dei files binari nel CATALOG. Un file binario è esattamente una copia esatta, bit per bit, delle informazioni caricate in un campo specificato delle posizioni di memoria di Apple. Queste posizioni possono essere contenute in un programma in linguaggio di macchina, in dati binari o in una "immagine" suddivisa in bytes dallo schermo per grafici ad alta risolvenza di Apple.

BSAVE

Il comando BSAVE crea un file denominato f e carica tutti i contenuti di un segmento di memoria.
La sintassi è

BSAVE f ,Aa, Lj [,Ss] [,Dd] [,Vv]

dove, come al solito, i parametri S, D e V stanno per il numero di slot, il numero di unità e il numero di volume. Notare che i parametri A e L non sono facoltativi.

Il parametro A specifica l'Indirizzo di partenza (in codice decimale o esadecimale) della parte di memoria da caricare sul disco. Un segno del dollaro ($) deve precedere un indirizzo espresso in esadecimale. Se il parametro A è minore di Ø o maggiore di 65535, viene presentato il messaggio.

SYNTAX ERROR

Pertanto, con questo comando non possono essere usati gli indirizzi negativi equivalenti. Nel campo da Ø a 65535, non viene generato alcun messaggio di errore se il parametro A specifica un indirizzo di partenza della memoria che non corrisponda ai chips di memoria effettivi installati. In pratica, non è utile specificare un parametro A maggiore dell'indirizzo massimo di memoria presente in Apple (49151 o $BFFF su un sistema a 48K).

Il parametro L specifica la lunghezza, in bytes, dalla parte di memoria da caricare. Se il parametro L è minore di Ø o maggiore di 65535, viene generato il messaggio

SYNTAX ERROR

Se il parametro L è Ø o nel campo da 32768 a 65535, viene generato il messaggio

RANGE ERROR

32767 è il numero massimo di bytes che possono essere caricati su disco in un singolo file. Se si desidera caricare più di 32767 posizioni di memoria, occorre usare più BSAVE. Nel campo da 1 a 32767, non viene generato alcun messaggio di errore se il parametro L specifica un campo di indirizzi di memoria, non tutti i quali corrispondono al chip effettivo di memoria installato. In pratica, non è utile specificare un campo di indirizzi di memoria che supera l'indirizzo massimo di memoria in Apple (49151 o $BFFF su un sistema a 48K).

Questi esempi creano ciascuno un file denominato PICTURE contenente un'immagine della seconda area dei grafici ad alta risolvenza della memoria di Apple. Questi sono operativamente identici, ma i rispettivi indirizzi di partenza e i parametri di lunghezza sono espressi in forme diverse.

```
BSAVE PICTURE, A$4ØØØ, L$2ØØØ
BSAVE PICTURE, A16384, L8192
BSAVE PICTURE, A16384, L$2ØØØ
BSAVE PICTURE, A$4ØØØ, L8192
```

BLOAD

Il comando BLOAD restituisce i contenuti di un file binario alla memoria di Apple II. BLOAD non cancella un programma BASIC in memoria, a meno che i dati non siano caricati (BLOAD) in quella particolare porzione di memoria contenente il programma.

La sintassi è

```
BLOAD f [,Aa] [,Ss] [,Dd] [,Vv]
```

dove S, D e V sono parametri come al solito. Se viene usato il parametro A, i contenuti del file binario sostituiscono una parte dei contenuti esistenti della memoria di Apple cominciando all'indirizzo a. Se non viene usato il parametro A, i contenuti dal file sono riportati alle stesse posizioni di memoria di Apple, i cui contenuti erano stati originariamente memorizzati (BSAVE).

Vedere BSAVE per una discussione completa del parametro A. Si supponga che il file binario PICTURE contenga un'immagine ad alta risolvenza. Uno o l'altro di questi esempi dispone l'immagine nella prima area dei grafici ad alta risolvenza della memoria di Apple :

```
BLOAD PICTURE, A8192
BLOAD PICTURE, A$2ØØØ
```

Nota : Un programma in linguaggio macchina può non essere più eseguibile se viene spostato ad una posizione di memoria diversa da quella nella quale era stato memorizzato.

BRUN

La sintassi del comando BRUN è la stessa di quella del comando BLOAD :

```
BRUN f [,Aa] [,Ss] [,Dd] [,Vv]
```

Il file binario f deve essere un programma in linguaggio macchina.
Innanzitutto BRUN esegue BLOAD. Se viene fornito il parametro A, i contenuti del file sono disposti nella memoria di Apple, iniziando dalla posizione a. Se non viene usato il parametro A, i contenuti del file sono riportati alle stesse posizioni della memoria di Apple, i cui contenuti erano stati originariamente memorizzati (BSAVE). Vedere BSAVE per una discussione completa del parametro A.

Dopo aver caricato il file (BLOAD), BRUN esegue un salto in linguaggio macchina (JMP) alla posizione a. Se il file era un programma in linguaggio di macchina, ciò dà inizio all'esecuzione di quel programma.

LA SUBROUTINE RWTS

Normalmente, l'accesso dell'utente a e dal DISK II è limitato all'uso del DOS. Per contro, ai programmatori in linguaggio macchina, è disponibile un altro metodo di accesso a DISK II. È quindi possibile saltare questo Capitolo se non si ha familiarità con il linguaggio macchina.

L'accesso a DISK II può avvenire direttamente dal linguaggio macchina, attraverso l'uso della subroutine RWTS, che fa parte del DOS. "RWTS" sta per "Read or Write a Track and Sector" (leggi o scrivi una pista e un settore).

Nella spiegazione che segue, qualsiasi numero preceduto da $ è un numero esadecimale.

Ogni disco inizializzato dall'unità DISK II è suddiviso in 35 piste, numerate da Ø a 34. Queste piste possono essere immaginate come solchi in un disco fonografico, salvo che non sono collegate l'una con l'altra. Fondamentalmente, le piste sono disposte in cerchi concentrici separati, con il grande foro al centro del mini floppy che forma il centro comune dei cerchi. La pista Ø è sul bordo esterno dal disco, mentre la pista 34 è la più vicina al centro. L'unità disco ha una "testina" che agisce più o meno come la puntina di un giradischi, salvo che la testina dell'unità disco è magnetica. Questa testina si sposta sulle diverse piste del disco, dove legge o scrive informazioni.

Ogni pista sul disco si compone di 16 settori. I settori sono raggruppamenti predefiniti su ciascuna pista, che consentono all'utente di lavorare con singoli blocchi di 256 bytes, anzichè con tutti i 4096 bytes che si possono inserire su una pista. I settori all'interno della pista sono singolarmente numerati, consecutivamente da Ø a 15 intorno al disco. Quando il disco gira, ciascun settore passa al di sotto della testina e in quel momento la testina può scrivere o leggere in quel settore. Ciascun settore si compone di due porzioni : il campo di indirizzo e il campo dei dati. Il campo di indirizzo contiene informazioni indicanti il numero della pista sulla quale è la testina, quale sarà il settore che passerà sotto la testina e il numero di volume del disco. Il campo dei dati contiene una forma codificata dei 256 bytes effettivi di dati che erano stati caricati su quel settore.

La subroutine "leggi o scrivi una pista e un settore" (detta anche subroutine RWTS) consente all'utente di scrivere o di leggere informazioni su qualsiasi pista e settore del disco, attraverso il linguaggio macchina. Per poter usare la subroutine RWTS, l'utente deve per prima cosa creare una tabella IOB (Blocco di controllo di Input/Output) ed una relativa Tabella delle caratteristiche della periferica. Le IOB dice alla subroutine RWTS in quale slot sarà e quale numero avrà l'unità disco, quale numero di volume aspettare sul disco, a quale pista è settore accedere e se leggere o scrivere sul disco. La tabella delle caratteristiche della periferica fornisce alla subroutine RWTS alcune informazioni necessarie per far funzionare il DISK II di Apple.

Per usare la subroutine RWTS, l'utente deve predisporre in qualche punto della memoria la IOB e la Tabella delle caratteristiche della periferica. Deve essere scritta e caricata in memoria una "subroutine di controllo". La subroutine deve eseguire un salto (JSR) all'indirizzo di partenza della subroutine RWTS (alla posizione $3D9). Quando avviene il salto alla subroutine RWTS, i registri A e Y devono contenere l'indirizzo della posizione di partenza dal IOB.

Il registro A deve contenere il byte di indirizzo, il registro Y il byte basso di indirizzo. Il formato dell'IOB è dato nella Tabella 3, al termine di questo paragafo. La Tabella 4 dà il formato della Tabella delle caratteristiche della periferica.

Segue ora un esempio del modo in cui usare la subroutine RWTS. L'IOB campione, le tabelle delle caratteristiche delle periferiche e la subroutine di controllo saranno tutti caricati nella memoria subito dopo la posizione $CØØ.

La sequente subroutine di controllo caricherà i registri A e Y con l'indirizzo della posizione di partenza dell'IOB e quindi salterà alla subroutine RWTS.

```
$CØØ- A9 ØC      LDA # $ØC    Carica il registro A con $ØC
$CØ2- AØ ØA      LDY # $ØA    Carica il registro Y con $ØA
$CØ4- 2Ø D9 Ø3   JSR $Ø3D9    Salta alla subroutine RWTS
$CØ7- 6Ø         RTS
$CØ8- ØØ         BRK
```

Il seguente IOB è quello che si usa per accedere alla slot 6, unità 1, per scrivere 256 bytes di memoria iniziando dalla posizione $2ØØØ sulla pista 18, settore 6 del disco:

Posizione	Codice	Scopo
$CØA	Ø1	Indicatore di tipo IOB, deve essere $Ø1
$CØB	6Ø	Numero di slot×16
$CØC	Ø1	Numero unità disco
$CØD	ØØ	Numero volume previsto
$CØE	12	Numero pista
$CØF	Ø6	Numero settore
$C1Ø	2Ø	Byte di ordine basso della Tabella delle caratteristiche della periferica
$C11	ØC	Byte di ordine alto della Tabella delle caratteristiche della periferica
$C12	ØØ	Byte di ordine basso dell'indicatore di partenza del buffer dei dati
$C13	2Ø	Byte di ordine alto dell'indirizzo di partenza del buffer dei dati
$C14	ØØ	Non utilizzato
$C15	ØØ	Non utilizzato
$C16	Ø2	Codice di comando, $Ø2=scrittura
$C17	ØØ	Codice di errore
$C18	ØØ	Numero effettivo di volume
$C19	6Ø	Numero di slot cui precedentemente si è avuto accesso
$C1A	Ø1	Numero di unità cui precedentemente si è avuto accesso

La seguente Tabella delle caratteristiche è obbligatoria e verrà inserita nella posizione $C2Ø, immediatamente dopo il IOB. Le posizioni $C1Ø e $C11 nel suddetto IOB puntano all'indirizzo della posizione di partenza delle Tabella delle caratteristiche delle periferiche.

Posizione	Codice	Scopo
$C2Ø	ØØ	Codice di tipo della periferica (inserire qui $ØØ)
$C21	Ø1	Numero delle fasi per pista (inserire qui $Ø1)
$C22	EF	Conta del tempo (inserire qui $EF)
$C23	D8	Conta del tempo (inserire qui $D8).

Dopo aver caricato il IOB in $CØA, la Tabella delle periferiche in $C2Ø e la subroutine di controllo per caricare i registri AY in $CØØ:

CØØG

oppure

CALL 3Ø72

per provocare l'esecuzione dell'intera routine.

TABELLA 3 : FORMATO DI IOB

Numero byte	Nome	Scopo
1	IBTYPE	Dice alla subroutine RWTS di quale tipo di IOB si tratta. Deve essere S01. Non sono definiti correntemente altri codici di tipo.
2	IBSLOT	Deve contenere il numero della slot (moltiplicato per 16) in cui è posta la scheda unità di controllo dell'unità disco. Ad esempio, se si desidera accedere alla slot n. 6, deve essere caricato in questa posizione il valore $60.
3	IBDRVN	Deve contenere il numero dell'unità disco cui accedere – $01 o $02.
4	IBVOL	Qui deve essere caricato il numero di volume del disco a cui accedere. Il volume assegnato a qualsiasi disco.
5	IBTRK	Qui deve essere caricato il numero della pista (da $ a 34) cui accedere. Deve essere compreso nel campo da $00a $22.
6	IBSECT	Qui deve essere caricato il numero del settore, (da $ a 15) cui accedere. Deve essere compreso nel campo tra $00 a $0F.
7 & 8	IBDCTP	Questi due bytes devono contenere l'indirizzo della posizione di partenza della Tabella delle caratteristiche della periferica (vedere più avanti). Il byte 7 deve contenere il byte di ordine basso dell'indirizzo, il byte 8 deve contenere il byte di ordine elevato.
9 & 10	IBBUFP	1 bytes 9 e 10 devono contenere l'indirizzo della posizione di partenza del "buffer dei dati". Il buffer dei dati è una sezione della memoria lunga 256 bytes sulla quale opererà la subroutine RWTS. Se si scrive sul disco, le informazioni del buffer dei dati saranno scritte sul disco. Se si sta leggendo dal disco, le informazioni provenienti dal disco saranno caricate in memoria nella posizione del buffer dei dati. 256 bytes è sia il minimo che il massimo delle informazioni che si possono leggere o scrivere mediante la subroutine RWTS.
11 & 12		Non utilizzato.
13	IBCMD	In questo byte è caricato il codice di comando che dice alla subroutine RWTS esattamente cosa fare. I valori che possono essere caricati nel byte 13 sono : $00 – – Comando nullo. Non fa nulla ma avvia l'unità disco e posiziona la testina. $01 – – Legge gli interi 256 bytes caricati sul disco sulla pista e sui settori specificati e li carica in memoria nella posizione del buffer dei dati. $02 – – Scrive i successivi 256 bytes caricati in memoria nella posizione del buffer di dati sul disco, sul settore e sulla pista specificati. $04 – – Esegue la formattazione del disco. Quando il disco è formattato, vengono scritti su ogni pista i semi-bytes autosincronizzanti. Poichè tutto il disco è formattato, i valori nei bytes 5 e 6 sono ignorati. Tutto il disco formattato è disponibile per l'uso; non c'è DOS o altro caricato sul disco fino a che l'utente non gli inserisce qualcosaltro.

14	IBSTAT	Questa posizione contiene il numero di codice per qualsiasi errore che si può incontrare durante l'esecuzione. Se la subroutine RWTS ritorna con un carry flag clear (flag del riporto spento) non si verifica alcun errore. Se ritorna con il carry flag set (flag del riporto acceso) questo byte indica il tipo di errore che si è verificato. $1Ø – – Il disco è protetto in scrittura e non può essere scritto. $2Ø – – Errore di accoppiamento di volume. Il numero di volume del disco trovato era diverso dal volume specificato nel byte 4. $4Ø – – Errore di unità. È successo qualcosa di insolito. $8Ø – – Errore di lettura. La routine RWTS non è stata in grado, dopo 48 tentativi ripetuti, di leggere il campo di indirizzo o il campo di dati. Se il campo dei dati per il settore specificato non è mai stato scritto, si ha un errore di lettura, in quanto non c'è niente da leggere.
15	IBSMOD	Il numero di volume del disco che è stato effettivamente trovato sarà caricato in questa posizione.

TABELLA 3 : FORMATO DI IOB (continuo)

Numero .Byte	Nome	Scopo
16	IOBPSN	Questo byte deve contenere il numero di slot (moltiplicato per 16) dello slot cui si è acceduto più recetemente. Ad esempio, se in precedenza si era avuto l'accesso ad una unità disco nello slot 5, bisogna caricare qui il valore $5Ø. Se non c'è unità di controllo nello slot specificato, il disco si blocca.
17	IOBPDN	Questo byte contiene il numero dell'unità disco cui si è avuto accesso più recentemente – $Ø1 o $Ø2.

Tabella 4 : FORMATO DELLA TABELLA DELLE CARATTERISTICHE DELLE PERIFERICHE

Numero byte	Nome	Scopo
1	DEVTPC	Codice del tipo di periferica, che dice di quale tipo di periferica si tratta. In questo byte deve essere caricato $ØØ per l'impiego con DISK II.
2	PPTC	Numero delle fasi per pista. Qui deve essere caricato $Ø1.
3 & 4	MONTC	Conta dei tempi, motore inserito, complementata, ad intervalli di 1ØØ microsecondi. Per impiego con DISK II nel byte 3 deve esserci $EF, e nel byte 4 $D8.



CAPITOLO 10

Input, output e concatenamento

Selezione dei dispositivi di I/O: IN #, PR # ED altri

Ci sono vari modi in cui le informazioni possono essere usate come input o output per il computer Apple. Molto spesso la tastiera serve come dispositivo di input. Solitamente Apple usa uno schermo per l'output, ma può essere usato qualsiasi accessorio o periferica collegato ad un'unità per l'input o per l'output usando i comandi IN# e PR#.

Esempi:

IN#6 Ottiene l'input successivo dalla periferica controllata dallo slot numero 6. Nota: se lot slot numero 6 contiene un'unità di controllo disco, questo comando provoca lancio del DOS. Se non c'è alcuna periferica collegata allo slot numero 6, il sistema può bloccarsi. Premere il tasto **RESET** per recuperare.
IN#Ø Ottiene l'input successivo dalla tastiera (non dallo slot numero Ø) invece che dalla periferica.
PR#1 Trasferisce l'output alla periferica controllata dallo slot numero 1, solitamente la stampante. Nota: se non è installata la scheda unità di controllo della periferica nello slot numero 1, il sistema può bloccarsi e occorre premere **RESET** per recuperare.
PR#Ø Ritorna l'output al video (e non allo slot numero Ø).

La sintassi per i comandi è

IN#s

oppure

PR#s

in cui s specifica lo slot da usare. Cosa succede dipende da s.

Valore di s	Risultato
Minore di Ø	SYNTAX ERROR
Ø	Ristabilisce l'abituale periferica (per IN#, input della tastiera, per PR#, output verso il video).
Da 1 a 7	Trasferisce alla periferica controllata dallo slot specificato (lancia il DOS se nello slot è presente l'unità di controllo del disco).
Da 8 a 16	SYNTAX ERROR nel modo ad esecuzione differita il sistema si blocca nel modo ad esecuzione immediata.
Da 17 a 65535	RANGE ERROR
Maggiore di 65535	SYNTAX ERROR

Il comando IN#Ø ristabilisce l'input dalla tastiera: PR#Ø ristabilisce l'output sul video.

Col DOS in atto, i comandi IN# e PR# possono essere usati nel modo solito esecuzione immediata (vedere i manuali BASIC).

Ma quando essi sono inseriti nelle righe di un programma, IN# e PR# devono prendere la forma dei comandi DOS, tipo ad esempio

```
10 D$=" ": REM CTRL-D
20 PRINT D$; "PR#1"
30 PRINT D$; "IN#2"
```

Quando il DOS non è in atto, i comandi IN# e PR# definiscono i contenuti dei registri di input e di output del Monitor di Apple per selezionare i dispositivi di input e di output desiderati.

Quando è in atto il DOS, i documenti dei registri di input e di output del Monitor di Apple sono predisposti per selezionare il DOS, mentre i contenuti dei registri di input e di output del DOS sono predisposti per selezionare i dispositivi di input e output desiderati. I seguenti paragrafi descrivono cosa succede ogniqualvolta un carattere lascia Apple o vi entra.

Quando Apple invia un carattere di output, il registro di output del Monitor Apple dirige quel carattere al DOS. Se il carattere deve essere inviato (in quanto non fa parte di un comando DOS), DOS esegue una rapida commutazione in due stadi:

1. Innanzitutto il DOS sostituisce i contenuti dei registri di input e di output del Monitor Apple e con i contenuti dei registri di input e di output del DOS, quindi invia il carattere alla periferica ora selezionata dei contenuti dei registri di input e output del Monitor Apple.
2. Successivamente, il DOS si ricollega ridisponendo i puntatori sul DOS nei registri di input e di output di Apple.

Analogamente, ogniqualvolta Apple chiede un carattere di input, il registro di input del Monitor Apple dirige quella richiesta al DOS.

Ancora una volta, il DOS esegue una commutazione rapida in due stadi:

1. Innanzitutto, il DOS sostituisce i contenuti dei registri di input e output del Monitor Apple con i contenuti dei registri di input e output del DOS, quindi ottiene un carattere di input dalla periferica ora selezionata dai registri di input e output del Monitor Apple.

2. Successivamente, il DOS si ricollega disponendo i puntatori verso il DOS nei registri di input ed output Apple.

Quando il DOS è in atto, intercetta tutti i caratteri di input dal dispositivo di input prima che raggiungano l'Applesoft o l'Integer BASIC o il Monitor. Ciò in quanto IN# e PR#, quando battuti sulla tastiera come comandi ad esecuzione immediata, possono essere, catturati e usati dal DOS per cambiare i registri di input e di output del DOS stesso.

Analogamente, il DOS intercetta tutti i caratteri di output da Apple, prima che essi raggiungano il dispositivo di output (ma **dopo** che hanno interessato i registri di input e di output del Monitor Apple). Ciò in quanto IN# e PR#, se emessi dall'interno di un programma ma non nei comandi DOS (PRINT), possono scollegare il DOS cambiando i registri di input e di output del Monitor Apple, prima che i comandi raggiungano il DOS.

Poichè gli interi contenuti dei registri di input e di output del Monitor Apple sono sostituiti ogni volta che il DOS tenta di inviare o di ricevere in carattere, il DOS solitamente si ricollegherà se non era stato collegato simultaneamente in entrambi i registri di input e di output.



Se si esegue un comando PR# dall'interno di un programma, con una riga di programma tipo

```
5Ø PR#1
```

il DOS verrà parzialmente scollegato e non sarà in grado di intercettare il successivo output. Il DOS è sempre collegato per l'input e il successivo tentativo di ottenere qualsiasi carattere di input provoca il ricollegamento del DOS sia all'output che all'input.

La stessa situazione si verifica con l'uso di IN# all'interno di programmi quando è in atto il DOS. Una riga di programma tipo

```
6Ø IN#1
```

scollegherà il DOS per il successivo input. Il DOS è ancora collegato per l'output e il successivo tentativo di trasmettere un carattere (anche un ritorno o un carattere di richiesta) provoca il ricollegamento del DOS sia per l'input che per l'output. Per evitare tali conflitti e consentire al DOS di gestire i registri di input e di output, emettere i comandi PR# e IN# nel modo ad esecuzione immediata, oppure come comandi DOS nelle righe di programma tipo quelle citate precedentemente:

```
1Ø D$="": REM CTRL-D
2Ø PRINT D$; "PR#1"
3Ø PRINT D$; "IN#2"
```

Il carattere **CTRL-D** dice al DOS che i successivi caratteri di output sono un comando DOS.

TABELLA 1: REGISTRI DI INPUT E DI OUTPUT DEL MONITOR APPLE

Registro di input del Monitor: Posizioni 56-57 ($38-$39)

Quando i contenuti del registro sono definiti da	Al valore	Il successivo input viene da
RESET Ø CTRL-K [Nota 1] IN#Ø [Nota 2]	– 741 ($FD1B)	Routine di input del Monitor dalla tastiera di Apple
s CTRL-L [Nota 1] IN#s [Nota 2] [dove s > Ø]	49152 + s*256 ($CsØØ)	Slot #s Se lo slot #s contiene l'unità di controllo del disco, lancia il DOS
Lancio del DOS	– 8574 + parte superiore della memoria (–$217E + $parte superiore della memoria)	DOS

Registro di output del Monitor: Posizioni 54-55 (§36-$37)

Quando i contenuti del registro sono definiti da	Al valore	Il successivo output va a
RESET Ø CTRL-P [Nota 1] PR # Ø [Nota 2]	–528 ($FDFØ)	Routine di output del Monitor al video
s CRTL-P [Nota 1] PR # s [Nota 2] [dove s > Ø]	49152 + s*256 ($CsØØ)	Slot # s Se lo slot#s contiene un'unità di controllo disco, lancia il DOS
Lancio DOS	– 8514 + parte superiore della memoria (–$2142 + $parte superiore della memoria)	DOS

Nota 1. I comandi s **CTRL-K** e s **CTRL-P** sono comandi del Monitor. Per battere **CTRL-K** (che non compare sul video, battere K mentre si tiene abbassato il tasto **CTRL**.

Nota 2. Quando è in atto il DOS, questo comando influisce sui contenuti del registro del Monitor Apple soltanto se il comando è emesso come istruzione in un programma memorizzato come istruzione PRINT **CTRL-D**.

Nota 3. Oltre ai comandi citati nella Tabella 1, l'inserimento (POKE) dei valori appropriati direttamente nelle posizioni di registro del Monitor Apple può anche essere usato per selezionare dispositivi di input e di output, oppure per ricollegare un DOS scollegato.

TABELLA 2: REGISTRI DOS DI INPUT E OUTPUT

Registro DOS di input

Quando i contenuti del registro sono definiti da	Al valore	Il successivo input viene da
Lancio DOS RESET 3DØG IN #Ø [Nota 4] PRINT D$;"IN#Ø" [Nota 5]	–741 ($FD1B)	Routine di input del Monitor dalla tastiera di Apple
IN# [Nota 4] PRINT D$;"IN#s" [Nota 5] [dove s >Ø]	49152 + s*256 ($CsØØ)	Slot #s Se lo slot #s contiene una unità di controllo del disco, rilancia DOS

Registro DOS di OUTPUT

Quando i contenuti del registro sono definiti da	Al valore	L'output successivo va a
Lancia DOS RESET 3DØG PR #Ø [Nota 4] PRINT D$;"PR#Ø" [Nota 5]	– 528 ($FDFØ)	Routine di output del Monitor al video
PR#s [Nota 4] PRINT D$;"PR#s" [Nota 5] [dove s >Ø]	49152 + s*256 ($CsØØ)	Slot #s Se lo slot #s contiene un'unità di controllo disco, rilancia il DOS

Nota 4. Quando è in atto il DOS, questo comando non influisce sui contenuti dei registri di input e di output del DOS se il comando è emesso come istruzione in un programma memorizzato e non in un'istruzione PRINT **CTRL-D**. Se viene eseguita una riga di programma tipo 12Ø PR#3
i contenuti del registro di output del Monitor Apple saranno cambiati, lasciando il DOS parzialmente scollegato fino al successivo input.

Nota 5. In questo comando, si presume che la variabile stringa denominata D$ sia stata assegnata al carattere di controllo D, o **CTRL-D**.
Questo carattere, che non compare sul video, è prodotto battendo D mentre si tiene abbassato il tasto **CTRL**.

Nota 6. Qualunque sia il dispositivo di output o di input scelto dai registri di input e output del DOS, l'input può anche essere ricevuto dal disco e l'output può anche essere inviato al disco.

Nota 7. Oltre ai comandi nella Tabella II, l'inserimento (POKE) diretto dei valori appropriati nelle posizioni di registro di input e di output del DOS, può anche essere usato per selezionare dispositivi di input e di output. Per contro, le posizioni specifiche di memoria dei registri di input e di output del DOS cambiano con i diversi formati di memoria del sistema e con le diverse versioni del DOS. Per questo motivo, esiste una speciale procedura per cambiare i contenuti delle posizioni di registro di input e di output del DOS. Si tratta di una procedura in due fasi:

a) Cambio delle posizioni dei registri di input e di output del Monitor Apple ai valori che si desidera contengano i registri di input e di output del DOS. (Ciò può essere fatto inserendo direttamente (POKE) le posizioni di registro del Monitor Apple oppure eseguendo istruzioni IN# o PR# non DOS in un programma memorizzato).

b) Eseguire CALL 1ØØ2 (dal Monitor, si deve battere $3EAG).
Dopo questo CALL (richiamo), il DOS verrà ricollegato attraverso i registri del Monitor Apple e i precedenti contenuti dei registri di input e di output del Monitor Apple compariranno nelle posizioni di registro di input e di output del DOS. Questa CALL può anche essere usata per ricollegare il DOS in qualsiasi momento il programma bisogni di scollegare temporaneamente il DOS. Vedere il programma a pagina 151 per un esempio dell'uso di questa tecnica.

Nota 8. I comandi del Monitor s **CTRL-K** o **CTRL-P** quando battuti sulla tastiera non sono riconosciuti dal DOS: essi interessano direttamente i registri di input e di output del Monitor Apple.

Concatenamento programmi integer BASIC

Talune applicazioni sono più facilmente implementate usando una serie di due o più programmi che vengono caricati (LOAD) ed eseguiti (RUN) sequenzialmente. In tali circostanze, il secondo programma deve spesso usare i valori delle variabili e delle matrici sviluppate del primo. Il solito comando RUN **cancella** le variabili e le matrici del primo programma, quando carica il secondo. In Integer BASIC (non in Applesoft), il comando DOS chain consente di caricare ed eseguire un secondo programma **senza** cancellare le variabili e le matrici del primo.

Si supponga di usare un programma Integer BASIC denominato PART ONE.
Il comando

```
CHAIN PART TWO
```

caricherà ed eseguirà il programma Integer BASIC denominato PART TWO senza cancellare i valori di qualsiasi variabile usata nel programma PART ONE. Il comando CHAIN può essere emesso nel modo ad esecuzione immediate come sopra indicato, oppure dall'interno delle ultime righe del programma PART ONE come un comando DOS

```
2ØØ1Ø D$="": REM CTRL-D
2ØØ2Ø PRINTS D$; "CHAIN PART TWO"
```

La sintassi per il comando è familiare:

```
CHAIN f [,Ss] [,Dd] [,Vv]
```

Concatenamento programmi Applesoft

Il comando CHAIN funziona soltanto con Integer BASIC, ma se non occorre trasferire le variabili è facile collegare i programmi Applesoft per caricare ed eseguire in sequenza. Nel primo programma basta comprendere un'ultima riga del tipo

```
2ØØØØ PRINT CHR$(4); "RUN PART TWO"
```

Quando questa riga viene eseguita, dà inizio al secondo programma (denominato PART TWO). Nel processo, il primo programma e tutte le sue variabili vengono cancellate.

Una diversa procedura deve essere usata per caricare ed eseguire una serie di programmi in Applesoft **senza** cancellare i valori precedenti delle variabili e delle matrici. Per concatenare in Applesoft, occorre usare un programma in linguaggio di macchina denominato CHAIN, presente sul disco System Master.

Per concatenare un programma denominato PART ONE ad un programma denominato PART TWO, occorre avere il programma CHAIN sullo stesso disco insieme al programma PART TWO (vedere la pagina successiva per le istruzioni). Quindi, basta inserire queste due righe come le ultime due da eseguirsi nel programma PART ONE.

```
2ØØØØ PRINT CHR$(4); "BLOAD CHAIN, A52Ø"
2ØØ1Ø CALL 52Ø"PART TWO"
```

Le due righe possono usare qualsiasi numero di riga, ma devono susseguirsi l'una dopo l'altra nel programma come indicato. La prima riga carica la capacità di concatenamento di Applesoft nel computer. La seconda riga esegue effettivamente il concatenamento: vedere, comunque, la successiva pagina per le avvertenze.



Non deve esserci alcuno spazio nella terza riga tra l'indirizzo CALL 52Ø e il seguente segno di virgolette (").
L'indirizzo CALL **non** deve essere espresso in esadecimale.

Se si dispone di Applesoft sulla scheda firmware in ROM, è possibile copiare il programma CHAIN su un altro disco come segue. Caricare innanzitutto il programma CHAIN nella memoria di Apple con il comando

```
BLOAD CHAIN, A12296
```

Quindi memorizzarlo sul disco desiderato, con il comando:

```
BSAVE CHAIN, A12296, L456
```



Notare che i parametri d'indirizzo per la copiatura di CHAIN sono uguali al parametro di indirizzo per l'uso effettivo di CHAIN.



APPENDICE A

Tipi di files usati con i comandi DOS

Salvo dove diversamente indicato, i comandi DOS possono essere usati sia nel modo ad esecuzione immediata che nel modo ad esecuzione differita all'interno di un programma. Per contro, alcuni comandi dei files di testo (ad esempio READ e WRITE) devono essere usati nel modo ad esecuzione differita.

La maggior parte dei comandi DOS si riferisce ad un file determinato, che può essere un file di testo (di dati) o un programma in Integer BASIC, in Applesoft, o in linguaggio macchina. Le tabelle che seguono indicano i tipi di file che possono essere usati da ciascun comando. La prima tabella elenca i comandi in ordine alfabetico, la seconda li raggruppa per tipo di file. I comandi CATALOG, FP, INT, MAXFILE, MON, NOMON, PR# e IN# non sono compresi in quanto non si riferiscono esplicitamente a files denominati.

FILES ELENCATI PER COMANDO DOS

Il comando DOS usa files:	File di programma Integer BASIC	File di programma Applesoft BASIC	File di testo ad accesso sequenziale	File di testo ad accesso casuale	File binario in linguaggio macchina
APPEND			x		
BLOAD					x
BRUN					x
BSAVE					x
CHAIN	x				
CLOSE			x	x	
DELETE	x	x	x	x	x
EXEC			x		
INIT	x	x			
LOAD	x	x			
LOCK	x	x	x	x	x
OPEN			x	x	
POSITION			x		
READ			x	x	
RENAME	x	x	x	x	x
RUN	x	x			
SAVE	x	x			
UNLOCK	x	x	x	x	x
VERIFY	x	x	x	x	x
WRITE			x	x	

Nota: Usare questi comandi solo nel modo ad esecuzione differita: APPEND, OPEN, POSITION, READ, WRITE.

Files elencati per tipo

SOLO FILES INTEGER BASIC:
CHAIN

FILES INTEGER BASIC O APPLESOFT:
INIT
LOAD
SAVE
RUN

SOLO FILES DI TESTO SEQUENZIALI:
APPEND
EXEC
POSITION

FILES DI TESTO AD ACCESSO CASUALE O FILES DI TESTO AD ACCESSO SEQUENZIALE:
OPEN
CLOSE
READ
WRITE

SOLO FILES IN LINGUAGGIO MACCHINA
BLOAD
BRUN
BSAVE

TUTTI I TIPI DI FILES
DELETE
LOCK
UNLOCK
RENAME
VERIFY

Nota: Questi comandi devono essere usati nel modo ad esecuzione differita: APPEND, OPEN, POSITION, READ, WRITE.

APPENDICE B

Messaggi DOS

Quando il DOS rileva un errore collegato all'uso del disco, normalmente presenta un messaggio che descrive l'errore e interrompe qualsiasi programma correntemente in esecuzione. Questi messaggi si aggiungono ai normali messaggi generati da Applesoft o Integer BASIC. I messaggi DOS possono essere distinti da quelli di Applesoft o Integer BASIC come segue:

Un messaggio Applesoft, tipo ad esempio

?SYNTAX ERROR

è preceduto da un punto interrogativo
Un messaggio Integer BASIC, tipo ad esempio

***SYNTAX ERROR

è preceduto da tre asteriscghi

Un messaggio DOS, tipo ad esempio

SYNTAX ERROR

non è preceduto da alcun carattere

Un messaggio DOS si presenta esattamente allo stesso modo sia che si operi in Applesoft, in Integer BASIC o in Monitor al momento in cui il messaggio viene generato.



Se si verifica un messaggio DOS quando si usa il Monitor, il sistema viene ripristinato nel BASIC dal quale si era in precedenza passati al Monitor.

Usando il comando ONERR GOTO di Applesoft (vedere il Manuale di Applesoft) si possono creare delle routines di gestione degli errori Applesoft, che agiscono sui messagi DOS che normalmente interromperebbero il programma. Quando si verifica un errore DOS dopo un comando ONERR GOTO in un programma Applesoft, nella posizione decimale di memoria 222 viene caricato un numero di codice per quel tipo di errore. Si tratta della stessa posizione di memoria in cui Applesoft memorizza il codice per un messaggio di errore Applesoft.
Il comando

Y = PEEK (222)

definisce il valore di Y al codice Applesoft ONERR GOTO corrispondente all'errore che ha provocato il salto ONERR GOTO Applesoft.

Qui di seguito, sono indicati i messaggi DOS e i loro corrispondenti codici ONERR GOTO Applesoft con la causa più comune di ciascun messaggio. Ciascuno dei messaggi viene quindi discusso, in maggiore dettaglio, insieme and un elenco più completo di cause e di rimedi.

CODICI ONERR GOTO

CODICE	MESSAGGIO DOS	CAUSA PIÙ COMUNE
1	LANGUAGE NOT AVAILABLE	Applesoft non presente su mini disco
2,3	RANGE ERROR	Parametro di comando troppo grande
4	WRITE PROTECTED	Linguetta di protezione di scrittura sul disco
5	END OF DATA	Lettura (READ) oltre la fine del file di testo
6	FILE NOT FOUND	File erroneamente battuto, o non su disco
7	VOLUME MISMATCH	Parametro di volume errato
8	I/O ERROR	Sportello aperto oppure disco non inizializzato (INIT)
9	DISK FULL	Troppi files su disco
1Ø	FILE LOCKED	Tentativo di sovrascrievre un file bloccato (LOCK)
11	SYNTAX ERROR	Nome di file, parametro o virgola errati
12	NO BUFFER AVAILABLE	Troppi files di testo OPEN (aperti)
13	FILE TYPE MISMATCH	Il file su disco non corrisponde al comando
14	PROGRAM TOO LARGE	Memoria Apple disponibile insufficiente
15	NOT DIRECT COMMAND	Il comando deve essere nel programma

DISCUSSIONE

LANGUAGE NOT AVAILABLE (linguaggio non disponibile)
(codice ONEER GOTO = 1)

Si verifica se il DOS non può trovare un linguaggio di programmazione, si tratti di Integer BASIC o di Applesoft, richiesto per eseguire un comando DOS. I comandi FP, INT, LOAD e **RUN** possono tutti iniziare una ricerca di linguaggio. Se è richiesto Integer BASIC, il DOS cerca quel linguaggio in ROM. Se è richiesto Applesoft, il DOS cerca prima il linguaggio in ROM, usando Applesoft da una scheda Applesoft ROM in firmware (se disponibile), indipendetemente dalla posizione dell'interruttore della scheda. Se nella ROM non viene trovato Applesoft, il DOS cerca sul minidisco nell'unità disco "presunta" – l'unità indicata dai valori presunti o più recenti dei parametri S e D. Il DOS non cerca su altre unità disco.

Questo messaggio si verifica solitamente dopo una richiesta DOS per un Applesoft su disco, se il disco nell'unità presunta non contiene il programma Applesoft, oppure usare il parametro D con qualsiasi comando DOS per selezionare un'altra unità. Un comando di questo tipo funziona perfettamente:

```
FP, D2
```

Se si pensa che il DOS debba trovare Integer BASIC in ROM, ma non lo trova, provare come segue:

1. Spegnere Apple e rimuovere il coperchio
2. Individuare la fila dei quattro grandi chips ROM (oggettini neri rettangolari) nel centro della scheda a circuito stampato principale. Questi chips sono contrassegnati "ROM F8", "ROM FØ", "ROM E8" e "ROM EØ.
3. Premere saldamente su questi chips
4. Rimontare il coperchio, riaccendere Apple e provare di nuovo INT.

Se si pensa che DOS debba trovare Applesoft sulla scheda ROM in firmware ma non lo trova, provare comme segue:

1. Spegnere Apple e rimuovere il coperchio
2. Scollegare la scheda ROM in firmware di Applesoft. Individuare la fila dei 5 grandi chips ROM (oggettini neri rettangolari) sulla scheda. Questi chips sono contrassegnati con: 1, 2, 3, 4 e 5 nella parte superiore e con DØ, D8, EØ, E8 e FØ nella parte inferiore.
3. Premere saldamente questi chips nelle rispettive prese.
4. Ricollegare di nuovo la scheda Applesoft nello slot n.Ø cioè nello slot più a sinistra.
5. Rimontare il coperchio, riaccendere Apple e provare di nuovo FP.

RANGE ERROR (errore di campo) (codice ONERR GOTO = 2 o 3)

Si verifica quando il valore di un parametro di comando DOS o di una quantità di comando DOS è troppo grande o troppo piccolo. Fare riferimento al Manuale per vedere quali comandi DOS sono usati e con quali parametri:

			Campo	
	Parametro	Lettere	Minimo	Massimo
Tutti i files:	Slot	S	1	7
	Unità	D	1	2
	Volume	V	Ø*	254
Files di testo sequenziale	Byte	B	Ø	32767
	Campo relativo	R	Ø	32767
	Campo assoluto (EXEC)	R	Ø	32767
Files di testo ad accesso casuale	Lunghezza record	L	1	32767
	Numero record	R	Ø	32767
Files binario	Indirizzo di inizio	A	Ø	65535
	Numero dei bytes	L	1	32767

		Campo	
Comando DOS	Quantità	Minimo	Massimo
PR# s	s	Ø	16**
IN# s	s	Ø	16**
MAXFILES n	n	1	16

* Il numero minimo di volume che INT assegnerà effettivamente ad un disco è 1.

** Il numero massimo di slots incorporati in Apple II è 7. Solo nel modo ad esecuzione differita, viene dato il messaggio SYNTAX ERROR per valori s da 8 a 16.

Nota: Non sempre l'uso di valori al di fuori dei campi suddetti provoca il messaggio RANGE ERROR. Qualsiasi parametro di comando o quantità di comando DOS minore di Ø o maggiore di 65535 provoca la comparsa di un messaggio SYNTAX ERROR e non il messaggio RANGE ERROR.

WRITE PROTECTED (protetto contro la scrittura) (codice ONERR GOTO = 4)

Si verifica quando DOS tenta di caricare informazioni sul disco, ma l'unità disco non rileva la tacca di "protezione di scrittura" o l'intaglio sul lato sinistro dell'involucro esterno del disco.

Quelle che seguono sono le cause più probabili.

1. C'è un'etichetta adesiva disposta sull'intaglio di protezione contro la scrittura del disco, per impedire l'accidentale sovrascrittura o la cancellazione di qualsiasi informazione sul disco. Questa etichetta può essere rimossa, dopodiché il DOS potrà memorizzare (SAVE o BSAVE) o scrivere (WRITE).

2. Sul disco non c'è intaglio di protezione contro la scrittura. Ciò vale per il disco System Master per avere la massima protezione.
Anche se non è consigliato, è tuttavia possibile ricavare accuratamente un intaglio, esattamente delle stesse dimensioni e nello stesso punto. Usare un intaglio di protezione di scrittura di un altro disco per modello.

3. Se compare questo messaggio mentre si esegue **(RUN)** il programma COPY e la causa non è fra le due precedenti, è possibile che il disco sia stato inserito nell'unità scorrettamente (in qualsiasi altra situazione il DOS emette il messaggio I/O ERROR per segnalare l'inserimento scorretto nell'unità e rileggere il Capitolo 1, a proposito dell'inserimento dei mini floppy.

END OF DATA (fine dei dati) (codice ONERR GOTO = 5)

Si verifica quando si tenta di richiamare informazioni dalla parte di un file di testo, in cui non sono mai state memorizzate informazioni. Qualsiasi byte oltre l'ultimo campo in un file di testo sequenziale od oltre l'ultimo campo di ciascun record di un file di testo ad accesso casuale può contenere il valore Ø. Lo Ø è il codice ASCII per un carattere nullo, un "nulla" e qualsiasi comando che provoca il richiamo di questo carattere dà luogo al messaggio END OF DATA. Ricordare che solo OPEN definisce automaticamente il puntatore della posizione nel file riportandolo all'inizio del file. Il messaggio compare solitamente dopo un commando INPUT o GET e può verificarsi in parecchi diversi modi:

1. Troppi INPUT successivi o INPUT con troppe variabili. Ciascun INPUT e ciascuna variabile INPUT provocano la lettura in Apple di un ulteriore campo adiacente.

2. Troppi GET successivi. Ciascun GET legge un byte o un carattere adiacente in più in Apple.

3. Il parametro B (per byte) era troppo grande. Nei files sequenziali, questo parametro non deve specificare un byte oltre l'ultimo carattere **RETURN** presente nel file. In un file ad accesso casuale, il parametro B non deve specificare un byte oltre l'ultimo carattere **RETURN** presente nel record correntemente selezionato.
Ricordare che il primo byte in un file o record è il byte Ø.

4. Il parametro R (per la posizione di campo relativa) in un comando POSITION era troppo grande. Nei files sequenziali, questo parametro non deve specificare un campo oltre l'ultimo esistente nel file. Nei files ad accesso casuale, il parametro R di POSITION non deve specificare un campo oltre l'ultimo esistente nel record correntemente selezionato.

Ricordare che il parametro R usato con POSITION non è lo stesso parametro R usato con READ. Esso specifica una posizione di campo nel file, rispetto alla posizione corrente del file e soltanto in avanti nel file.

RØ specifica nessun cambiamento nella posizione corrente del file. R1 salta alla posizione di file davanti al primo byte che segue il campo che contiene la posizione corrente.

POSITION analizza in avanti i contenuti del file, byte per byte, cercando il carattere **RETURN** Rp-esimo. Se incontra un byte Ø (il carattere nullo) prima di trovare il carattere **RETURN** richiesto, viene emesso immediatamente il messaggio END OF DATA; non è necessario in effetti eseguire INPUT o GET del carattere nullo.

5. Il parametro R (per posizione assoluta di campo) in un comando EXEC era troppo grande. Questo parametro può specificare il primo campo oltre l'ultimo esistente in un file, ma il tentativo di specificare il secondo campo oltre l'estremità del file provoca il messaggio END OF DATA. Ricordare che RØ specifica il primo campo in un file.

6. Il parametro R (per record) in un comando READ specificava il record di file ad accesso casuale in cui non era stato memorizzato nulla. Prima di poter leggere (READ) da un particolare record in un file ad accesso casuale, occorre per prima cosa scrivere (WRITE) qualche informazione in quel record.

Ricordare che il parametro R di READ **non è lo stesso** parametro R usato da POSITION o da EXEC. Il parametro R di READ specifica un record assoluto in un file: RØ è il **primo** record del file e cosi via.

Il DOS usa il parametro L del comando OPEN per il calcolo del punto d'inizio del record r-esimo, cosicchè OPEN che precede READ **deve** usare lo stesso parametro L del commando OPEN che precedeva WRITE per quel file.

FILE NOT FOUND (file non trovato) (codice ONERR GOTO = 6)

Si verifica quando taluni comandi DOS specificano il nome file che non è nel CATALOG del mini floppy nell'unità disco selezionata o presunta. Pertanto, i comandi SAVE, BSAVE e OPEN possono **creare** un file il cui nome non esisteva precedentemente. Oltre a questi, CLOSE può essere usato con qualsiasi nome valido. Un nome file specificato da qualsiasi altro comando DOS deve già esistere sul disco. Questo messaggio può verificarsi per varie cause:

1. È possibile che sia stato impostato erroneamente il nome del file, a causa di un errore di battitura o omettendo la virgola che separa il nome file dal parametro seguente.

Controllare il CATALOG per l'esatta ortografia del nome del file.

Attenzione: se sono stati accidentalmente battuti caratteri di controllo nel nome di un file, CATALOG non li presenta. Vedere "Nomi file" nell'Appendice F.

2. Il file è su un altro disco. Controllare il CATALOG.

3. Il file è stato accidentalmente cancellato (DELETE). Controllare il CATALOG.

4. Quando si usa il comando INIT, si specifica un nome file che il DOS successivamente tenta di eseguire (RUN) ogniqualvolta si lancia il sistema con quel disco nell'unità 1. A meno che non si scriva un programma BASIC e lo si memorizzi, usando il nome dato a INIT, il messaggio FILE NOT FOUND sarà emesso ogniqualvolta il sistema è lanciato con quel disco nell'unità 1. Se non è possibile ricordare il nome di questo programma "greeting", basta aggiornare il mini floppy, con il programma Master Create.

VOLUME MISMATCH (errore di volume) (codice ONERR GOTO = 7)

Si verifica quando il parametro volume (V) usato in un comando DOS non corrisponde al numero di volume assegnato al disco nell'unità selezionata o presunta, quando quel disco è stato inizializzato (INIT). Il numero di volume di un disco è indicato in testa a ciascun CATALOG. A meno che un comando DOS non specifichi un particolare volume, il numero di volume del disco è **ignorato** e non viene emesso alcun messaggio. Se un comando DOS specifica il volume Ø, il numero di volume del disco è ancora **ignorato**. Se con INIT non viene dato alcun numero di volume oppure se viene dato il numero di volume Ø, il disco viene inizializzato al numero di volume presunto 254.

I/0 ERROR (error di I/O) (codice ONERR GOTO = 8)

Si verifica dopo un tentativo non coronato da successo di caricare dati su disco o di richiamare dati da un disco (DOS tenta 96 volte, quindi getta la spugna). Questo messaggio può verificarsi nei modi seguenti:

1. Lo sportello dell'unità disco selezionata o presunta è aperto. Chiudere lo sportello dell'unità disco.

2. Nell'unità disco selezionata o presunta non c'è il mini floppy. Inserire un disco nell'unità e chiudere lo sportello.

3. Il mini floppy nell'unità selezionata o presunta non è stato inizializzato (INIT). Inizializzare il mini floppy (INIT).

4. Il mini floppy è inserito scorrettamente. Controllare il disco e rileggere il paragrafo nel Capitolo 1 sull'inserimento del mini floppy.

5. Durante l'esecuzione di un comando VERIFY il DOS ha trovato che il file specificato non era caricato correttamente sul mini floppy. Se l'informazione del file è ancora in memoria, tentare di caricarla di nuovo (meglio se un disco diverso).

6. Il parametro D (Drive) del comando DOS ha specificato un'unità disco che non esiste nel sistema. L'unità presunta è ora quella non esistente. Per risolvere l'errore, specificare semplicemente il corretto parametro D con successivo comando DOS.

7. Il parametro S del comando DOS ha specificato uno slot che non contiene una scheda di controllo del disco nel sistema.



Ma possono sorgere problemi. Lo slot presunto è ora lo slot vuoto che l'ultimo comando DOS ha specificato. Il successivo comando DOS senza un parametro di slot andrà allo slot vuoto e provocherà lo stesso messaggio precedente. Peggio ancora, il DOS pensa che l'unità disco che non esiste in quello slot sia tuttora in funzione. Il successivo comando DOS che specifica lo slot corrretto invierà il sistema in un limbo permanente in attesa che l'unità non esistente si fermi, prima di passare all'unità recentemente selezionata! Si deve in tal caso o rilanciare il sistema (perdendo naturalmente qualsiasi programma in memoria) oppure:
a) Premere il tasto **RESET** (quindi il sistema si blocca)
b) Battere CALL-151, quindi 3 DØG (il sistema è ora di nuovo OK)
c) Battere CATALOG, Ss (dove s = slot corretto)

DISK FULL (disco pieno) (codice ONERR GOTO = 9)

Si verifica quando il DOS tenta di caricare informazioni sul disco e trova che su quel disco non è disponibile altro spazio di memoria. Si possono riempire fino a 496 settori con informazioni caricate dall'utente, come presentate da CATALOG (se un singolo file supera 255 settori, la presentazione CATALOG della sua lunghezza inizia di nuovo ØØØ). Se compare il messaggio DISK FULL, si può stare sicuri che tutti i files sono chiusi (CLOSE) e che il DOS ha memorizzato tutto quello che poteva (escludendo una certa parte del file dal disco). Se si riceve questo messaggio mentre si memorizza un file denominato STUFF, la prima cosa da farsi è di:

DELETE STUFF

e quindi memorizzare il programma su un altro mini floppy che abbia più spazio disponibile.

Se si riceve il messaggio DISK FULL e si tenta quindi immediatamente di memorizzare (SAVE, BSAVE) o scrivere (WRITE) qualsiasi file su disco, prima di averne cancellato qualsiasi altro (DELETE), la lunghezza in settori di un file indicato nel CATALOG sarà definita a Ø. Non preoccuparsi: nonostante l'aspetto strano del CATALOG, il file è corretto. Possono anche verificarsi altri eventi strani. Per evitare tali situazioni, se compare un messaggio DISK FULL, cancellare (DELETE) alcuni files prima di tentare di memorizzarne altri.

FILE LOCKED (file bloccato) (codice ONERR GOTO = 1Ø)

Si verifica quando si tenta di memorizzare (SAVE, BSAVE) di scrivere (WRITE) o di cancellare (DELETE) usando un nome file che è stato bloccato sul disco (LOCK), che si trova nell'unità presunta o selezionata. Controllare il CATALOG: i nomi dei files bloccati (LOCK) sono preceduti da un asterisco (*) nel video del CATALOG. Un file viene bloccato (LOCK) per impedire una sovrascrittura accidentale. Usare un altro disco oppure sbloccare (UNLOCK) il file desiderato.

SINTAX ERROR (errore di sintassi) (codice ONERR GOTO = 11)

Si verifica quando il DOS incontra un errore di sintassi in un comando DOS. Controllare il Manuale per l'esatta sintassi richiesta per il commando in questione. Il problema potrebbe essere un nome file non valido (vedere Appendice F), un simbolo di parametro scorretto, un parametro mancante, un separatore mancante o scorretto (solitamente una virgola). Questo messaggio compare anche se un valore di parametro o una quantità di commando è un numero negativo o è maggiore di 65535, oppure nel caso dei comandi IN# o PR# usati nel modo ad esecuzione differita, se lo slot specificato va da 8 a 16.
Raramente, ogni comando DOS provoca un messaggio Syntax Error in Applesoft o Integer BASIC. Ciò solitamente significa che il DOS non è stalo lanciato o è "scollegato" dall'input e dall'output. Provare, premendo il tasto **RESET**, quindi, rilanciare il disco.

NO BUFFERS AVAILABLE (buffers non disponibili) (codice ONEER GOTO = 12)

Si verifica quando un comando DOS richiede un altro buffer di files per l'input o per l'output e tuti i buffers di file disponibili sono già in uso. Al lancio del sistema, il DOS riserva sufficiente spazio nella memoria di Apple per tre buffers dei files di input e di output. Un successivo commando MAXFILES piò aumentare o diminuire il numero di buffers dei files disponibili e un comando CLOSE può sbloccare il bufffer dei files correntemente in uso per i files di testo.

Molti comandi DOS usano un buffer dei files per l'input o l'output durante la loro esecuzione e quindi abbandonano quel buffer al termine dell'esecuzione del comando.

Quando viene aperto un file di testo (OPEN), ad esso viene assegnato un buffer dei files per l'input e l'output. Questo buffer rimane in uso, generalmente fino a che il relativo file non è chiuso (CLOSE) sia specificamente da un nome file che da un CLOSE senza nome, che disalloca tutti i buffers dei files di testo. Un file di testo non viene automaticamente chiuso da un programma che arriva alla fine. Per conservare spazio nel buffer, chiudere (CLOSE) i files non appena si è terminato di usarli. Ricordare che il successivo OPEN è ripristinato all''inizio del file dal puntatore di posizione nel file.



Il comando MAXFILES può essere usato per aumentare lo spazio di buffer prima di scrivere il programma o di caricare il programma in memoria. L'aumento di MAXFILES sposta HIMEM in basso, e può cancellare le righe di programma memorizzate o le stringhe Applesoft. La variazione di MAXFILES nel mezzo di un programma può essere particolarmente pericolosa.

FILE TYPE MISMATCH (tipo di file non corrispondente) (codice ONERR GOTO = 13)

Si verifica quando un comando DOS tenta di usare un nome di file che è già assegnato ad un file il cui tipo è inadatto al comando presente. Se si è sicuri che il comando è corretto, usae un nome file che non si trova sul disco, usare un disco diverso, ridenominare (RENAME) o cancellare (DELETE) il file esistente.

Questo messaggio si produce da parecchie diverse combinazioni scorrette di comandi DOS con tipi di files esistenti. Ecco le combinazioni corrette:

LOAD f, RUN f, SAVE f	f deve essere un file di programma Applesoft o Integer BASIC
CHAIN f	f dece essere un file di programma Integer BASIC
OPEN f, READ f, WRITE f, APPEND f, POSITION f, EXEC f	f deve essere un file di testo
BLOAD f, BRUN f, BSAVE f	f deve essere un programma binario o un file di dati

Il nome file del programma "greeting", specificato con INIT o UPDATE, deve fare riferimento ad un file di programma Applesoft o Integer BASIC.

PROGRAM TOO LARGE (programma troppo grande) (codice ONERR GOTO = 14)

Si verifica quando un comando DOS tenta di caricare un file su disco nella memoria di Apple e trova la memoria disponibile insufficiente per contenere l'intero file. Può verificarsi che un precedente programma o l'operatore abbiano definito HIMEM troppo basso per il compito corrente o che un grande MAXFILES abbia definito HIMEM troppo basso. Se si definisce il numero di buffers dei files 3, usando il comando
MAXFILES 3

HIMEM viene riportato al valore di lancio indicato nell'appendice D, Tabella 2.



Se si è in Integer BASIC e HIMEM è definito basso (ad esempio, per proteggere la memoria dello schermo ad alta risolvenza), si possono incontrare problemi nel passaggio all'Applesoft su disco. Applesoft su mini floppy occupa circa 12,5K di memoria, ma uno spostamento al l'Applesoft su disco (con FP o LOAD o RUN7 **non** ripristina HIMEM al massimo. Quando il DOS tenta di caricare il programma Applesoft dal disco, viene emesso il messaggio PROGRAM TOO LARGE se HIMEM è al di sotto di 131ØØ. Il sistema sarà lasciato ancora in Integer BASIC e si deve definire HIMEM più alto da Integer BASIC. Vedere Appendice D, Tabella 2 per il valore massimo di HIMEM per il sistema DOS e con 3 buffers dei files.



Nel decidere se un programma si inserisce o meno nella memoria disponibile, il DOS cerca soltanto il numero dei **settori** di disco occupati dal programma. In generale, il programma non riempie completamente l'ultimo settore (256 bytes) ma il DOS ignora questo fatto e confronta soltanto il byte di ordine elevato di LOMEM (Integer BASIC) o di HIMEM (Applesoft) con il byte di ordine elevato della posizione di fine programma prevista. Pertanto, un programma che si adattasse in memoria, ma che lasciasse meno di 256 byte di memoria liberi dopo il caricamento, potrebbe provocare un messaggio PROGRAM TOO LARGE. Talvolta, ciò può essere corretto spostando leggermente HIMEM o LOMEM per cambiare il byte di ordine elevato, prima di caricare il programma.

NOT DIRECT COMMAND (assenza di comando diretto) (codice ONERR GOTO = 15)

Si verifica quando si tenta di usare uno dei comandi dei files di testo APPEND, OPEN, POSITION, READ o WRITE dal modo ad esecuzione immediata. Questi comandi DOS possono essere usati soltanto dall'interno di istruzioni PRINT nelle righe di programma.

APPENDICE C

Formato delle informazioni su mini floppy

Questa Appendice spiega come le informazioni vengono caricate su un disco e come il DOS ricorda il punto in cui una particolare informazione è stata memorizzata.

Nella discussione che segue, il segno del dollaro ($) o la label "Hex' che precedono il numero indicano che il numero stesso è espresso in esadecimale.

Uno sguardo al processo di memorizzazione

Nel sistema Disk II, le informazioni sono registrate si disco in 35 zone concentriche denominate **piste** (tracce). Queste piste sono numerate da $ØØ, la pista più esterna, fina a $22, la pista più interna. La testina di lettura e di registrazione dell'unità disco può essere spostata verso l'interno, per fermarsi ed abbassarsi sopra ciascuna di queste 35 diverse zone dal disco in rotazione.

Inoltre, la lunghezza di ciascuna pista sul disco è divisa in 16 segmenti, denominati **settori**. Questi settori sono numerati da $Ø a $F, e in ciascuno di essi possono essere caricati fino a 256 ($1ØØ) bytes di informazione. Una volta che la testina di lettura e di registrazione dell'unità disco è posizionata su una data pista, i sedici settori di quella pista passano sotto la testina, uno dopo l'altro, ogniqualvolta il disco ruota. Il DOS registra sempre le informazioni sul disco a gruppi di 256 bytes, riempiendo esattamente un settore ogni volta.

Per memorizzare le informazioni sul disco, il DOS inserisce dapprima 256 bytes (cioè il valore di un settore) di informazioni in una zona della memoria di Apple denominata **buffer dei files**. Quando questo buffer dei files è pieno, le informazioni vengono caricate in un settore sul disco. Quindi DOS riempie il buffer dei files di Apple con i successivi 256 bytes di informazioni e le memorizza sul disco.

In generale, il DOS inizia a caricare un programma o un file di testo ogniqualvolta può trovare sul disco un settore inutilizzato. Quando quel settore è riempito con i suoi 256 bytes di informazione, il DOS trova un altro settore libero, magari su un'altra pista, e continua a registrarvi le informazioni. Questo processo continua fino a che l'intero file non è stato caricato.

Per ricordare quali settori e quali piste contengono le informazioni per un particolare file, il DOS esegue un elenco di ciascuna pista e settore usati, man mano che carica il file, quindi, carica quell'elenco denominato **elenco di pista/settore**, in un altro settore (o settori) ancora libero sul disco.

Infine, il nome dei files, il tipo di file, la lunghezza in settori e la posizione sul disco dell'elenco pista/settore del file sono registrati in una zone speciale della pista $11 denominata catalogo. A questo punto, anche **la mappa dei bit della pista** del disco viene aggiornata per mostrare correttamente quali sono i settori di ciascuna pista correntemente in uso.

Scrittura di un file di testo sequenziale

Le entrate in un file di testo si compongono da 1 a 32767 caratteri memorizzati come i rispettivi equivalenti codici ASCII e terminati da un carattere **RETURN** (ASCII $Ød o ASCII $8D).

Ciascuna di tali entrate è denominata "**campo**".

In un file di testo sequenziale (nessun parametro di lunghezza è stato specificato al momento dell'apertura) i campi sono caricati immediatamente l'uno dopo l'altro (vedere Capitolo 6). Il DOS scrive il primo byte di ciascun nuovo campo immediatamente dopo il carattere **RETURN** che ha terminato il campo precedente (a meno che diversamente istruito da un parametro byte). Ogniqualvolta il file viene aperto (OPEN) il DOS dimentica la posizione corrente all'interno del file e ricomincia a scrivere (WRITE) di nuovo nel byte Ø (a meno che non sia stato diversamente istruito da un parametro byte).

Per riscrivere un particolare campo o carattere all'interno di un file sequenziale, WRITE può essere usato con il parametro B (per byte) per iniziare a scrivere al byte **assoluto** specificato del file (il primo byte nel file è il byte Ø, il successivo byte è il byte 1, ecc.) Il byte specificato può essere prima o dopo la posizione corrente nel file.



È molto difficile ricordare esattamente quale carattere compare in ogni byte di un file di testo, particolarmente in un file di testo sequenziale. Per questo motivo, l'uso del parametro byte nei files di testo sequenziali non è consigliato.

Il comando POSITION può essere usato con un parametro R (per campo relativo) per spostare un puntatore in **avanti** (soltanto) nel file di un numero specificato di campi **relativi** a tale posizione corrente nel file. Una porzione di programma tipo

```
12Ø  PRINT D$; ''OPEN NAMES''
13Ø  PRINT D$; ''POSITION NAMES, R1
     3''
14Ø  PRINT D$; ''WRITE NAMES''
16Ø  PRINT ''APPLE COMPUTER''
17Ø  PRINT D$; ''CLOSE NAMES''
```

tenterà di scrivere (WRITE) (i caratteri APPLE COMPUTER nel file NAMES, iniziando dal primo byte del 14° campo (il primo campo è il campo relativo Ø).



POSITION può spostare al primo byte di un dato campo relativo alla posizione corrente in un file di testo sequenziale. Se si riscrive quindi quel campo (re-WRITE) per contro, ci si deve assicurare il ribattere **esattamente** in quel campo (PRINT) lo stesso numero di caratteri originariamente (PRINT). Battendo meno caratteri (PRINT) si creano due nuovi campi: il campo immediatamente stampato o battuto (PRINT) e l'estremità di coda del campo originale, che è stata sovrascritta. Battendo più caratteri (PRINT) di quelli contenuti nel campo originale, si sovrascrivono alcuni dei caratteri all'inizio del campo successivo.

Scrittura in un file di testo ad accesso casuale

Per un file di testo ad accesso casuale, viene specificato un parametro Length (lunghezza) quando il file viene aperto (OPEN). Il parametro Length determina il numero dei bytes in un **record**, che è un campo o un gruppo di campi che il DOS tratta come un'unità. Ciascun record in un file di testo ad accesso casuale è come un file di testo sequenziale separato, la cui lunghezza totale massima è stata specificata dal parametro Lenght. Fintanto che si rimane all'interno della lunghezza massima, è possibile scrivere (WRITE) e riscrivere (re-WRITE) tutto ciò che si desidera, senza interessare qualsiasi altro record nel file. WRITE può essere usato con i parametri R (per record) e B (per byte) per iniziare a scrivere in un qualsiasi byte di un record specificato.



Dal momento in cui **qualsiasi** comando DOS termina la scrittura (WRITE) non è possibile usare POSITION per saltare in avanti in campi diversi all'interno del record specificato dal comando WRITE.

Il DOS usa il parametro Length per calcolare dove scrivere il primo byte di ciascun nuovo record (L bytes oltre il primo byte del record precedente). Il DOS semplicemente salta qualsiasi byte tra l'ultimo carattere del record precedente e il byte L. I bytes saltati continuano a contenere qualsivoglia valore era caricato in un tempo precedente (vedere il successivo paragrafo per i particolari).



Se si tenta di scrivere (WRITE) **più** caratteri in un record ad accesso casuale di quanti non ne siano stati specificati nel parametro Lenght, tutti i caratteri vengono caricati correttamente sul disco. Per contro, quando si inizia a scrivere (WRITE) nel successivo record, il DOS calcola la posizione di partenza del nuovo record, come se il record precedente fosse stato compreso nella lunghezza specificata. Il nuovo record pertanto sovrascrive gli ultimi caratteri del precedente record maggiorato, compreso il carattere **RETURN** di marcatura di fine dell'ultimo campo del record precedente. Il risultato è abbastanza confuso.

Come il DOS scrive nei files di testo: procedura generale

Quando si scrive (WRITE) un campo in un file di testo, il DOS controlla per prima cosa sul disco se ci sono o meno informazioni già memorizzate nel settore che dovrebbe contenere quel campo. Se il file non ha mai usato prima d'ora quel settore, il DOS inserisce degli zeri in tutti i 256 bytes di un buffer dei files Apple e quindi consente di inserire le informazioni in quel buffer per la successiva memorizzazione, nel corretto settore del disco. Il contenuto del buffer dei files è caricato sul disco quando le informazioni hanno completamente riempito i 256 bytes del buffer, oppure quando il file viene chiuso (CLOSE).

Pertanto, Quando si scrive su un particolare settore **per la prima volta**, ai bytes non utilizzati viene attribuito il valore zero. Un tentativo di leggere (READ) un byte contenente uno Ø (il codice ASCII per il carattere nullo) fa comparire il messaggio

END OF DATA

Ma se il DOS trovache il file ha già delle informazioni nel settore che dovrebbe contenere il campo sul quale si sta ora scrivendo, legge tutti i 256 bytes da quel settore nel buffer dei files di Apple. Dopo aver cambiato uno qualsiasi di quei bytes del buffer dei files per contenere la nuova informazione – i comandi WRITE, POSITION (solo per i files sequenziali) e PRINT se ne prendono cura – il DOS, quindi, rimemorizza i contenuti del buffer esattamente nello stesso settore di disco in cui avevano avuto origine. I contenuti del buffer dei files sono rimemorizzati sul disco, quando si tenta di cambiare qualsiasi byte non nel settore che era stato letto nel buffer dei files, oppure quando il file viene chiuso (CLOSE).



Pertanto, se si scrivono (WRITE) altre informazioni per un file e il DOS memorizza quelle informazioni in un settore di disco già usato dal file, non si riscrivono degli zeri nei bytes non utilizzati. Uno qualsiasi di quei bytes di settore che non sono stati riscritti continuerà a contenere qualsivoglia informazione che possa essere stata caricata prima del comando WRITE. Ciò vale per i bytes inutilizzati al termine di un file di testo sequenziale e vale anche per i bytes non utilizzati in ciascun record di lunghezza fissa di un file di testo ad accesso casuale.

Contenuto dei settori dei file

Ora che si conosce il processo generale per registrare un file su un disco, si può discutere ciascun elemento in maggior dettaglio. Le informazioni effettive memorizzate, settore per settore, sono diverse per ciascun tipo di file.

FORMATO DEI SETTORI DEI FILES
per i diversi tipi di file

Tipo di file	Settore	Byte (esadecimale)	Contenuti del byte
BASIC (entrambi i tipi)	1. settore	Ø	Lunghezza programma, byte basso
		1	Lunghezza programma, byte alto
		da 2 a FF	Programma simbolizzato
	Settori successivi	Tutti i bytes	Programma simbolizzato
TESTO	Tutti i settori	Tutti i bytes	Rappresentazione ASCII del testo: un byte/carattere (SØØ contrassegna la fine del file)

Tipo di file	Settore	Byte (esadecimale)	Contenuti del byte
BINARIO	1. settore	Ø	Indirizzo di partenza RAM, byte basso
		1	Indirizzo di partenza RAM, byte alto
		2	Lunghezza della parte RAM, byte basso
		3	Lunghezza della parte RAM, byte alto
		da 4 a FF	Dati binari
	Settori successivi	Tutti i bytes	Dati binari

L'elenco traccia/settore

Quando un file viene memorizzato sul disco, il DOS esegue un elenco delle posizioni di disco usate dal file. Questo elenco settore/pista viene, quindi, caricato sul disco allo stesso modo in cui è stato caricato il file stesso. I contenuiti di un elenco di pista/settore sono i seguenti:

1. settore di un elenco pista/settore

Byte (esadecimali)	Contenuti dei byte
Ø	Non usato
1	Link: numero di pista dove si può trovare la continuazione dell'elenco pista/settore
2	Link: numero settore dove si può trovare la continuazione dell'elenco pista/settore (se entrambi i bytes di link=Ø, nessun link)
da 3 a 4	Non usati
5 - 6	Numero base di settori (conta gruppi di 122 settori)
da 7 a 8	Non usati
C	Numero di pista del primo settore di file
D	Numero di settore del primo settore di file
E	Numero di pista del secondo settore di file
F	Numero di settore del secondo settore di file
1Ø	Numero di pista del terzo settore di file
11	Numero di settore del terzo settore di file
•	• •
•	• •
•	• •
FE	Numero di pista del 122esimo settore di file
FF	Numero di settore del 122esimo settore di file

Se qualsiasi coppia di pista/settore è ØØ, ciò indica un settore non assegnato (solitamente alla fine del file, quantunque i files di testo possono contenere indicatori ØØ per molti settori non ancora assegnati dove possono essere scritti bytes o records futuri).

I successivi settori dell'elenco pista/settore (se l'elenco si estende oltre le 122 coppie di pista/-settore) sono identic al primo sopra descritto, salvo che le coppie di pista/settore si riferiscono ai successivi gruppi dei 122 settori dei files. Inoltre, i bytes Link 1e 2 saranno differenti per ciascun successivo settore. Ciascuna coppia Link dà al DOS la posizione di disco della successiva porzione dell'elenco pista/settore. Se entrambi i bytes di Link sono Ø, ciò indica la porzione finale dell'elenco pista/settore.

Con un file di testo, soltanto le coppie pista/settore per quel settore effettivamente contenenti informazioni compaiono diverse da Ø nell'elenco pista/settore. Il DOS calcola la posizione corretta per la coppia pista/settore all'interno dell'elenco, riempiendo le coppie pista/settore non assegnate con zeri.

Pertanto, se il parametro length per un file ad accesso casuale è 128 (due records per settore) e si scrive (Write) soltanto fino al numero di record 27ØØ, vengono effettivamente usati soltanto tredici settori di mini floppy: uno per i contenuti del record numero 27ØØ e dedici settori per l'elenco pista/settore. I contenuti dei records dal numero Ø al numero 2683 possono talvolta occupare 1342 settori: ma solo se sono stati fisicamente tutti seritti (write).

Il catalogo del mini floppy

Su ogni disco inizializzato (INIT) la pista $11 è riservata per le informazioni che riguardano i contenuti del disco. Questo è il punto in cui il DOS memorizza l'elenco contenente, per ciascun file, il nome file, il tipo di file, il numero di settori occupati dal file (MOD 256) e la posizione nel disco dell'elenco di pista/settore del file. Il comando CATALOG provoca la comparsa della maggior parte di queste informazioni sul video. Ciascun settore di un elenco di disco è formattato come segue:

Un settore di un elenco disco

Byte (esadecimale)	Contenuti del byte
Ø	Non usato
1	Link: numero di pista dove si può trovare la continuazione dell'indirizzario (normalmente $11)
2	Link: numero di settore in cui si può trovare la continuazione dell'elenco (se entrambi i bytes di Link=Ø, nessum link)

Byte (esadecimale)	Contenuti del byte	
da 3 a A	Non usato	
da B a 2D	Entrata indirizzario per il file 1	(vedere più avanti)
da 2E a 5Ø	Entrata indirizzario per il file 2	
da 51 a 73	Entrata indirizzario per il file 3	
da 7a a 96	Entrata indirizzario per il file 4	
da 97 a B9	Entrata indirizzario per il file 5	
.da BA a DC	Entrata indirizzario per il file 6	
da DD a FF	Entrata indirizzario per il file 7	

I numeri di file indicati per le sette entrate dell'indirizzario sono arbitrari. Quando un file viene cancellato (DELETE), il DOS **contrassegna** l'entrata dell'indirizzario per quel file (vedere Tabella che segue). La successiva volta in cui viene caricato un file, il DOS sostituisce la vecchia entrata di indirizzario contrassegnata con l'entrata dell'indirizzario per il nuovo file. Pertanto, mentre il DOS originariamente riempie l'indirizzario nell'ordine indicato, le cancellazioni dei files rendono quest'ordine privo di significato.

L'indirizzario del disco inizia alla pista $11 settore $F. Se occorre più spazio per caricare altre entrate dell'indirizzario, il settore $F viene concatenato (link) al settore $E. Se occorre ancora altro spazio, il settore $E viene concatenato (link) al settore $D, e cosi via, fino al settore $1. Ciò consente all'indirizzario di caricare entrate per un massimo di 105 files diversi.

Ciascuna entrata di indirizzario è scritta nel seguente formato:

ENTRATA DI INDIRIZZARIO PER UN FILE

Byte relativo (esadecimale)	Contenuti del byte
Ø	Numero di pista dell'elenco di pista/settore del file (cambiato in $FF quando il file viene cancellato)
1	Numero di settore dell'elenco pista/settore del file
2	Tipo di file (vedere discussione alla pagina successiva)
da 3 a 2Ø	Nome file
21 - 22	Conta del settore: il numero dei settori del disco (MOD 256) occupati dal file.

Un **byte relativo** dell'entrata della directory specifica ciascun byte all'interno dell'entrata, quantunque ciascuna entrata inizi ad un numero di byte effettivo diverso all'interno del settore dell'indirizzario. Per trovare un byte di settore assoluto corrispondente ad un byte relativo, aggiungere il byte relativo al primo byte assoluto di settore dell'entrata (come indicato nella precedente tabella).

Poichè viene usato soltanto un byte per caricare una conta di settori di file, la conta massima di settore di indirizzario è 255 ($FF). Se un file supera 255 settori, la sua conta di settore (come è presentata da CATALOG) inizia di nuovo a ØØØ. Ciò non influisce sull'uso del file, ma può dare un'impressione errata di quanto pieno sia il disco.

Agli 8 bits di un byte indicante il tipo di file, il byte n. 2 in un'entrata di indirizzario di un file (vedere la tabella precedente) sono assegnati i valori come segue:

Byte indicante il tipo di file

Bit	Simbolo CATALOG	Tipo di file designato
7	*	Il file è bloccato (protetto in scrittura) se questo bit=1 Il file è sbloccato (non protetto) se questo bit=Ø
6		Tipo espansione per uso futuro (normalmente zero)
5		Tipo espansione per uso futuro (normalmente zero)
4		Tipo espansione per uso futuro (normalmente zero)
3		Tipo espansione per uso futuro (normalmente zero)
2	B	File binario se questo bit=1
1	A	File Applesoft BASIC se questo bit=1
Ø	I	File Integer BASIC se questo bit=1
	T	File di testo se i bits da Ø a 6 sono tutti zero

Questo tipo di file è determinato da un bit 1 che compare in uno dei bits da Ø a 6. Se i bit da Ø a 6 sono tutti Ø il file è presunto come file di testo. Il bit di designazione di tipo di un file può pertanto assumere i seguenti valori:

Valori per il byte che indica il tipo di file		
	Valore del byte di tipo (esadecimale)	
Tipo di file	File sbloccato	File bloccato
Testo	Ø	8Ø
Integer	1	81
Applesoft	2	82
Binario	4	84

Tabella dei volumi

Il settore $Ø della pista $11 contiene la tabella dei volumi degli indici del disco o VTOC.

TABELLA DEI VOLUMI DEGLI INDICI (VTOC)
Pista $11, settore $Ø

Byte (esadecimale)	Valore (esadecimale)	Descrizione
Ø	2	Non usato
1	11	Numero di pista del primo settore dell'indirizzario
2	ØF	Numero di settore del primo settore dell'indirizzario
3	4	Numero di release DOS
4	Ø	Non usato
5	Ø	Non usato
6	da 1 a FE	Numero di volume del disco (presunto:$FE)
da 7 a 26	Ø	Non usato
27	7A	Numero massimo di coppie di pista/settore possibili in ciascun settore di un elenco pista/settore
da 28 a 2F	Ø	Non usato
3Ø	FF	Questi 4 bytes sono una maschera per le mappe dei bits di pista (vedere le successive due pagine): ciascun bit 1 abilita uno dei 16 settori da utilizzarsi in ogni pista
31		
32	FF+FF	
33	ØØ	
	ØØ	

(continua alla pagina successiva)

Byte (esadecimale)	Valore (esadecimale)	Descrizione	
34	23	Numero delle piste per disco	
35	ØF	Numero dei settori per pista	
36	ØØ	Numero dei bytes per settore: byte basso	
37	Ø1	Numero dei bytes per settore: byte alto	
da 38 a 3B	Ø	Mappa dei bits pista Ø	(Ma queste piste
da 3C a 3F	Ø	Mappa dei bits pista 1	non sono disponibili
da 4Ø a 43	Ø	Mappa dei bits pista 2	all'utente)
da 44 a 45	?	Mappa dei bits pista 3	
da 46 a 47	Ø	Mappa dei bits pista 3	
48 e 49	?	Mappa dei bits pista 4	
4A e 4B	Ø	Mappa dei bits pista 4	
•	•		•
•	•		•
•	•		•
78 e 79	?	Mappa dei bits pista $1Ø	
7A e 7B	Ø	Mappa dei bits pista $1Ø	
da 7C a 7F	Ø	Mappa dei bits pista $11 (elenco e VTOC)	
8Ø e 81	?	Mappa dei bits pista $12	
82 e 83	Ø	Mappa dei bits pista $12	
•	•		•
•	•		•
•	•		•
CØ e C1	?	Mappa dei bits pista $22	
C2 e C3	Ø	Mappa dei bits pista $22	
da C4 a FF	Ø	Non usato	

Mappa dei bits delle tracce

Partendo dal byte $38 del VTOC (vedere tabella precedente) i successivi gruppi di 4 bytes contengono ciascuno **una mappa dei bits di pista** per una delle 35 piste del disco. La disposizione dei bits 1 e dei bits Ø all'interno di una mappa dei bits della pista, mostra al DOS quali settori di quella pista del disco sono correntemente in uso e quali settori sono liberi. La mappa dei bits per ciascuna pista usa il seguente formato:

MAPPA A BITS DELLE TRACCE
Per ogni traccia del disco

Byte	Bit	Settore designato (Esadecimale)	Byte	Bit	Settore designato (Esadecimale)
1.	7	F	2.	7	7
	6	E		6	6
	5	D		5	5
	4	C		4	4
	3	B		3	3
	2	A		2	2
	1	9		1	1
	Ø	8		Ø	Ø
il 3. e 4. di riserva					

Se un bit in una mappa dei bits di pista contiene il valore 1, il settore corrispondente a quel bit è libero. Se un bit nella mappa contiene il valore Ø, il settore corrispondente a quel bit è correntemente in uso. Se i bits contrassegnati "riserva" nella tabella che precede contengono i valori Ø, non sono usati. La mappa dei bits di pista per una tipica pista potrebbe comparire come qui di seguito:

MAPPA TIPICA DEI BITS DI PISTA

Primo byte	Secondo byte	Terzo byte	Quarto byte
Ø Ø Ø Ø Ø Ø 1 1	1 1 1 1 1 1 1 1	Ø Ø Ø Ø Ø Ø Ø Ø	Ø Ø Ø Ø Ø Ø Ø Ø
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑	↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑		
F E D C B A 9 8	7 6 5 4 3 2 1 Ø	Riserva	Riserva
Settori designati		Non usati	

1= Settore libero (supponendo il bit corrispondente della maschera, VTOC bytes da $3Ø fino a $33, sia a sua volta 1)
Ø=settore in uso

Quando un file viene caricato su un disco (usando WRITE, SAVE o BSAVE), al file viene immediatamente assegnata una intera pista (dove possibile) e la mappa dei bits della pista mostra l'intera pista in uso. Pertanto, quando il file viene chiuso (CLOSE) quei settori non effettivamente usati sono di nuovo considerati liberi, nella mappa dei bits per quella pista.



I settori effettivamente usati per caricare un'informazione di un file, per contro, possono essere definiti liberi soltano quanto il nome file è cancellato (DELETE). Si supponga ad esempio che il disco contenga un file BASIC da 1ØØ settori denominato BIG. Se si vuole ora memorizzare (SAVE) sullo stesso disco un file ad un settore con lo stesso nome BIG (sovrascrivendo il vecchio file), un CATALOG del disco rivelerà che il file a due settori BIG sta sempre usando i 1ØØ settori. Per liberare i settori non usati da un file BASIC denominato BIG, usare la seguente sequenza di comandi:

```
LOAD BIG
DELETE BIG
SAVE BIG
```

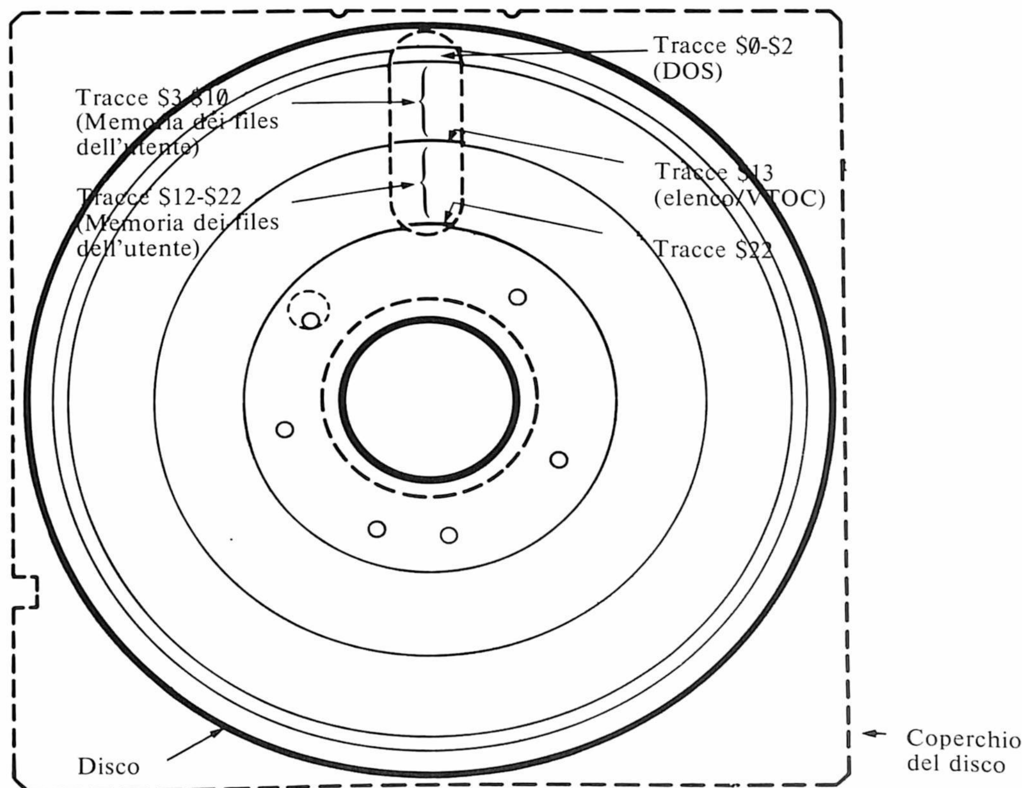
Un processo simile può essere usato per sbloccare settori non necessari usati da files binari.

Per sbloccare settori non necessari usati in un file di testo, occorre leggere (READ) ciascuno dei campi del file in Apple. Se si caricano tutti i campi in una matrice, si può quindi cancellare (DELETE) il file originale, prima di scrivere (WRITE) di nuovo ciascun record sul disco usando il nome file originale. Un altro modo per fare ciò è di leggere ciascun campo in Apple e immediatamente riscrivere (WRITE) il campo sul disco usando un nome file diverso da quello originale. Una volta letto e riscritto l'ultimo campo, si può cancellare (DELETE) il file originale.

Ordine di allocazione di tracce e settori

Ciascun disco contiene 35 piste, 3 delle quali sono riservare per il DOS e 1 per l'indirizzario, lasciando quindi 31 piste per l'utente. Ciascuna pista contiene 16 settori, cosicchè tutti i 31*16 ossia 496 settori disponibili all'utente.

I settori sono riempiti partendo dal settore $F e tornando indietro al settore $Ø. le piste sono riempite partendo con la pista $12 (immediatamente all'interno della pista indirizzario/V-TOC) e proseguendo verso l'interno fino alla pista $22 (la pista più interna). Quando la pista $22 è stata riempita, si prosegue con la pista $1Ø (appena all'esterno della pista indirizzario/V-TOC) e lavorando verso l'esterno in direzione della pista $3 (la pista più esterna disponibile per l'utente).



ORDINE DI ALLOCAZIONE DELLE PISTE

	Riempito per primo	Riempito per ultimo
Primo:	$12	$22
Quindi:	$1Ø	$Ø3

ORDINE DI ALLOCAZIONE DEI SETTORI

Riempito per primo	Riempito per ultimo
$F	$Ø

Richiamo di informazioni dal disco

Per richiamare un file dal disco, il DOS segue il processo usato per caricare il file, ma alla rovescia. Dopo un comando del tipo

LOAD FILE

oppure

BLOAD FILE

ad esempio, il DOS va sull'indirizzario dei files del disco nella pista $11, e trova l'entrata di indirizzario contenente il nome file, che contiene anche la posizione sul disco (per piste e settori) dell'elenco di pista/settore del file desiderato. Il DOS quindi va a questo elenco pista/settore e legge la prima coppia pista/settore, che specifica la posizione sul disco del primo settore contenente il programma denominato FILE. Quando il DOS ha letto quel primo settore del programma in Apple, ritorna all'elenco del settore/pista per la posizione del secondo settore del programma e così via.

Lettura da un file sequenziale

Ad esempio, leggendo da un file di testo sequenziale, con una parte di programma tipo

```
5ØPRINT D$; "READ TEXTFILE"
6Ø INPUT A$
```

il processo originale è identico a quello descritto per il caricamento (LOAD) di un file di programma, con la sola differenza che il settore contenente il successivo campo del file di testo (tutti i caratteri della posizione corrente nel file fino al successivo carattere **RETURN**) vengono letti nel buffer dei files di Apple, in risposta al comando INPUT. Di consequenza, i bytes effettivi di settore che costituiscono il campo desiderato sono assegnati alla variabile A$. Questo processo viene ripetuto se il campo si estende a più di un settore di disco. Ciascun successivo comando input provoca la lettura del file per riprendere dal buffer dei files di Apple, se esso già contiene il campo appropriato, oppure per leggere un altro settore di disco in Apple. Ciò continua fino a che non è stato letto l'ultimo campo o fino a che qualche comando non chiude (CLOSE) il file.

Usando il comando READ con il parametro B (per byte) è possibile far si che il successivo input inizi a leggere dal byte assoluto specificato nel file (il primo byte del file è Ø, il successivo è 1, ecc.). Questo byte può essere prima o dopo la posizione corrente all'interno del file. Per usare questo parametro efficacemente, comunque, occorre conoscere i contenuti di ogni byte nel file. Il comando POSITION usa il parametro R (per campo relativo), per spostare in avanti (soltanto) il puntatore della posizione corrente di DOS del numero specificato di campi nel file rispetto alla posizione corrente. Ogniqualvolta si apre un file (OPEN) il DOS dimentica la sua posizione corrente nel file stesso ed inizia di nuovo a leggere (READ) dall'inizio del file (a meno che non sia istruito diversamente da un parametro byte).



Il comando INPUT tratta una risposta in modo alquanto diverso in Integer BASIC e in Applesoft. Se compaiono nel campo di risposta taluni caratteri, tipo ad esempio i due punti o la virgola, gli ulteriori caratteri nel campo possono essere ignorati o assegnati a variabili multiple INPUT (se presenti). Per i particolari, vedere l'appropriato Manuale per Integer BASIC o per Applesoft.

Lettura da un file ad accesso casuale

Il processo di lettura del testo è alquanto diverso quando si legge da un elenco specificato di un file di testo ad accesso casuale (vedere anche SCRITTURA IN UN FILE AD ACCESSO CASUALE in questa Appendice). In un file di testo ad accesso casuale, ciascun record è composto dallo stesso numero di bytes, specificati nel parametro Length al momento dell'apertura del file e prima di scrivere nel file stesso. Quando questo stesso file viene aperto prima di leggerlo (READ) viene dato un parametro identico a Length. Per ritrovare l'inizio di un particolare record (specificato dal parametro R del comando READ) il DOS usa il parametro Length per calcolare i numeri dei bytes da tutti i records precedenti. Quel numero è quindi diviso per 256 ($1ØØ per determinare quanti settori di file il DOS deve saltare per raggiungere il settore contenente il record desiderato. Quindi, il DOS esamina l'elenco pista/settore del file e trova la posizione di disco del settore di file desiderato. Infine, il DOS legge il corretto settore nel buffer dei files di Apple. A questo punto, i bytes corretti possono essere letti dal buffer dei files.



Questo stesso processo di richiamo verrebbe eseguito se il file di testo fosse stato originariamente caricato come **file sequenziale** o come file ad accesso casuale usando una lunghezza completamente diversa. Il DOS calcola ciecamente la posizione di byte e di settore del record richiesto, secondo qualsivoglia parametro Length specificato al momento dell'apertura del file prima della sua lettura, **indipendentemente** dal parametro Length (se previsto) che era stato usato nello scrivere il file la prima volta.

Usando il comando READ con entrambi i parametri R (per record) e B (per **byte**) è possibile fare sì che il successivo INPUT inizi a leggere dal byte assoluto specificato nel **record** specificato (il primo byte di ciascun record è Ø, il successivo è 1, ecc.). Questo byte può essere prima o dopo la posizione corrente all'interno del record. Per usare questo parametro efficacemente, comunque, si deve conoscere il contenuto di ogni byte nel record specificato.

Il comando POSITION, anche se principalmente inteso per l'accesso a **files sequenziali** può essere usato con un parametro R (per campo relativo) per spostare soltanto **in avanti** il puntatore della posizione corrente del DOS del numero specificato di campi nel record corrente, rispetto alla posizione corrente nel record. READ viene usato con il parametro R (per record, stavolta) per spostare il puntatore della posizione corrente all'inizio del record desiderato.

L'uso di POSITION cancella il modo READ (senza ripristinare il puntatore di posizione) e un altro READ (stavolta senza parametro) ripristina il modo READ.

Ogniqualvolta si apre un file, il DOS dimentica la posizione corrente nel file e riprende a leggere all'inizio del file (salvo dove diversamente istruito da un parametro byte e/o record).

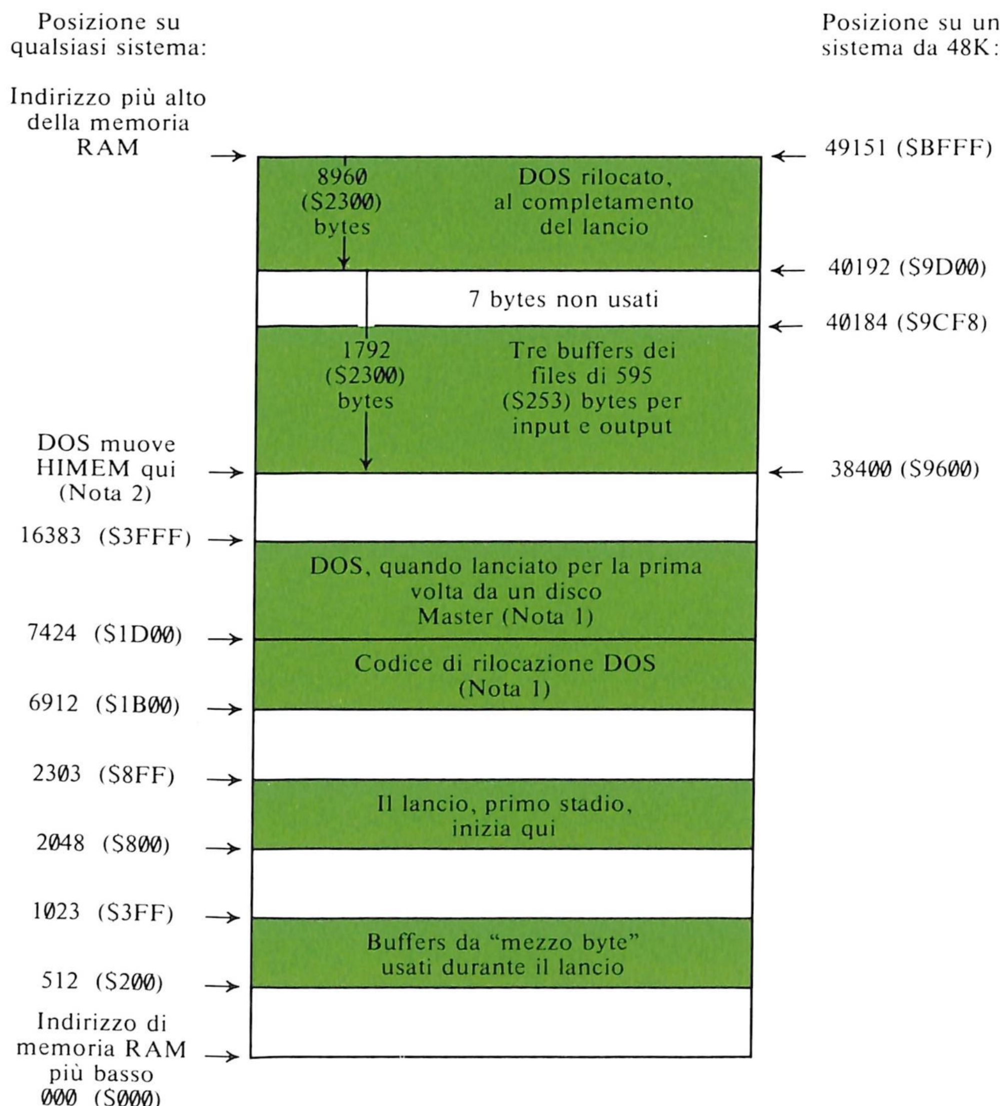


Il DOS non conserva alcuna informazione per quanto riguarda la struttura, il formato, la lunghezza dei records o la lunghezza di campo dei files di testo. Per usare efficacemente i files di testo ad accesso casuale, occorre osservare le dettagliate informazioni scritte sulla struttura di questi files, oppure mantenere le informazioni all'inizio del file.

APPENDICE D

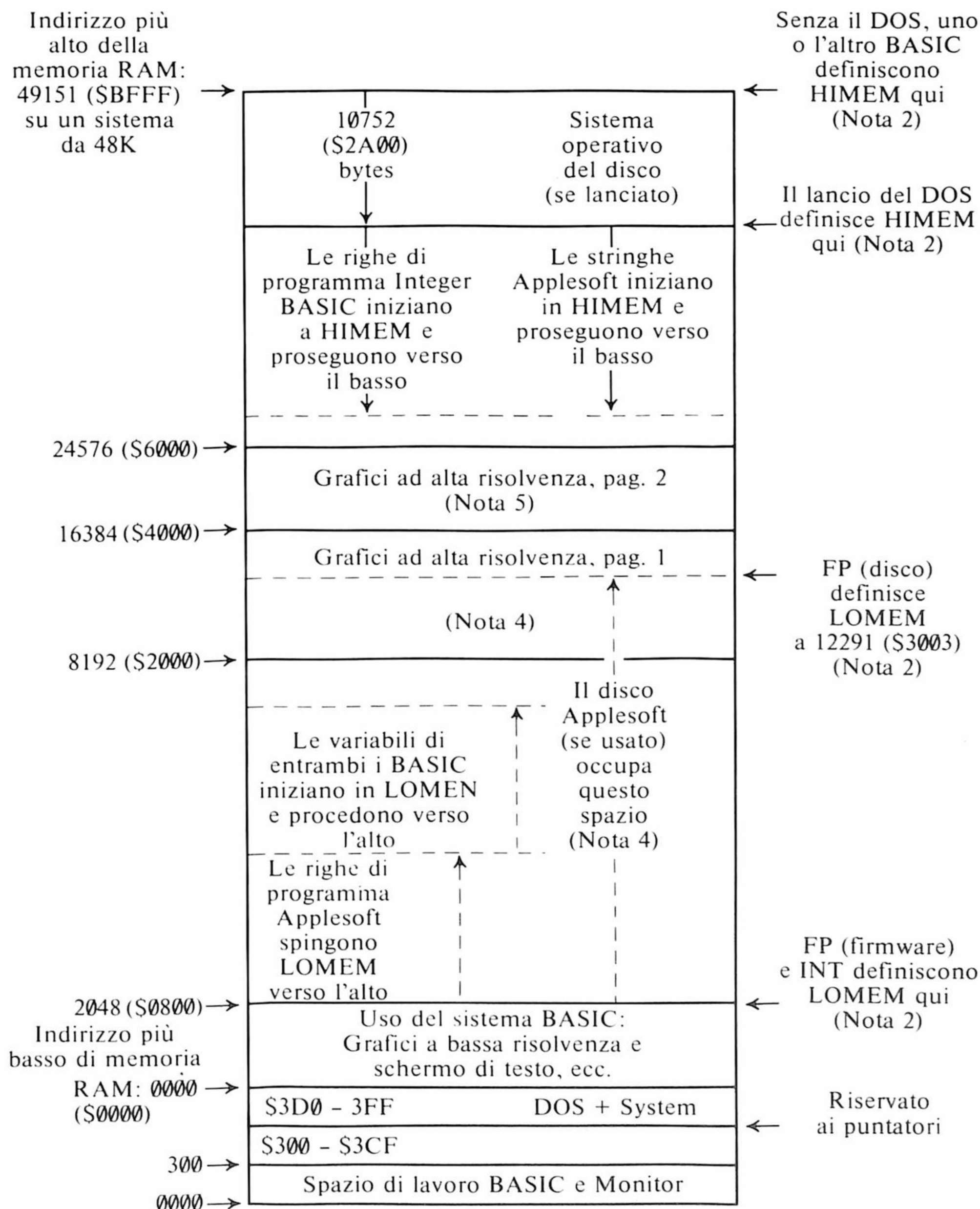
Uso della memoria

Tabella 1 : mappe della memoria di apple II
A. Zone di memoria occupate al lancio del DOS



Nota 1. Questa zona di memoria non viene influenzata lanciando un disco secondario : il DOS è posto direttamente al di sotto dell'indirizzo della memoria RAM più alto disponibile sul sistema che ha inizializzato (INIT) il disco secondario, sia esso appropriato o meno al sistema presente.

B. Zone di memoria usate dal DOS e dall'uno e dall'altro BASIC



Nota 2. Se il sistema è Integer BASIC, il puntatore HIMEM si può trovare (iniziando con il byte basso, quindi col byte alto) nelle posizioni 76-77 ($4C-$4D). Se il sistema è in Applesoft BASIC, il puntatore HIMEM è nella posizione 115-116 ($73-$74) stesso formato. Vedere Tabella 2 per il valore di HIMEM definito al lancio del DOS. Aumentando MAXFILES si sposta HIMEM verso il basso di altri 595 bytes per ciascun buffer dei files aggiunti. Per le posizioni di altri puntatori di programma Applesoft, consultare il Manuale di programmazione Applesoft II BASIC, Appendice I.

Tabella 2 : valore di HIMEM definito al lancio del DOS

Quando il DOS viene lanciato, HIMEM viene definito secondo la quantità di memoria nel sistema :

Dimensione del sistema	Indirizzo RAM più alto		HIMEM : definito al lancio di DOS		
	Decimale	Esadecimale	Decimale	Esadecimale	
16K	16383	$3FFF	5632	$16ØØ	
2ØK	2Ø479	$4FFF	9728	$26ØØ	
24K	24575	$5FFF	13824	$36ØØ	
32K	32767	$6FFF	22Ø16	$56ØØ	
36K	36863	$8FFF	26112	$66ØØ	
48K	49151	$BFFF	– 27136	$96ØØ	(Nota 3)

Nota 3. Il numero – 27136 potrebbe anche essere scritto 384ØØ ma Integer BASIC non accetta numeri superiori a 32767. In Integer BASIC, gli indirizzi di memoria maggiori di 32767 deveno essere espressi coi rispettivi equivalenti negativi. L'equivalente negativo di qualsiasi indirizzo decimale positivo n è (n – 65536).

Nota 4. Usando i grafici ad alta risolvenza, la pagina 1, cancella i contenuti delle posizioni di memoria da 8129 à 16383. A meno che DOS definisca HIMEM ad un valore maggiore di 16383, un tentativo di usare la pagina 1 dei grafici ad alta risolvenza cancella una parte del DOS. Ciò significa che non è possibile usare contemporaneamente Disk II e i grafici ad alta risolvenza, a meno che il sistema non contenga almeno 32K di memoria.

Con Applesoft su disco, un tentativo di usare la pagina 1 dei grafici ad alta risolvenza cancella una parte di Applesoft. Con Applesoft su disco, è possibile usare soltanto la pagina 2 dei grafici ad alta risolvenza, se il sistema contiene almeno 36K di memoria. Vedere Nota 5.

Nota 5. L'uso della pagina 2 dei grafici ad alta risolvenza cancella i contenuti delle posizioni di memoria da 16384 a 24575. A meno che il DOS non definisca HIMEM ad un valore maggiore di 24575, un tentativo di usare la pagina 2 dei grafici ad alta risolvenza può cancellare una parte del DOS. Ciò significa che non è possibile usare contemporaneamente Disk II e la pagina 2 dei grafici ad alta risolvenza, a meno che il sistema non contenga 36K di memoria.

APPENDICE E

Punti di entrata DOS e schemi

Punti di entrata DOS

Routine per ricollegare DOS (se la pagina 3 viene sovrascritta) :

Dimensioni del sistema	Indirizzo decimale (CALL)	Indirizzo esadecimale (G)
48K	−25153	$9DBF
32K	23999	$5DBF
16K	7615	$1DBF

Il comando Minitor 3DØL presenta questo numero nella parte superiore destra.

Posizioni che contengono l'indirizzo di partenza e la lunghezza di un programma caricato (BLOAD) :

Dimensioni del sistema	Indirizzo di partenza (byte basso)		Lunghezza del programma (byte basso)	
	Decimale	Esadecimale	Decimale	Esadecimale
48K	43634	$AA72	43616	$AA6Ø
32K	2725Ø	$6A72	27232	$6A6Ø
16K	1Ø866	$2A72	1Ø848	$2A6Ø

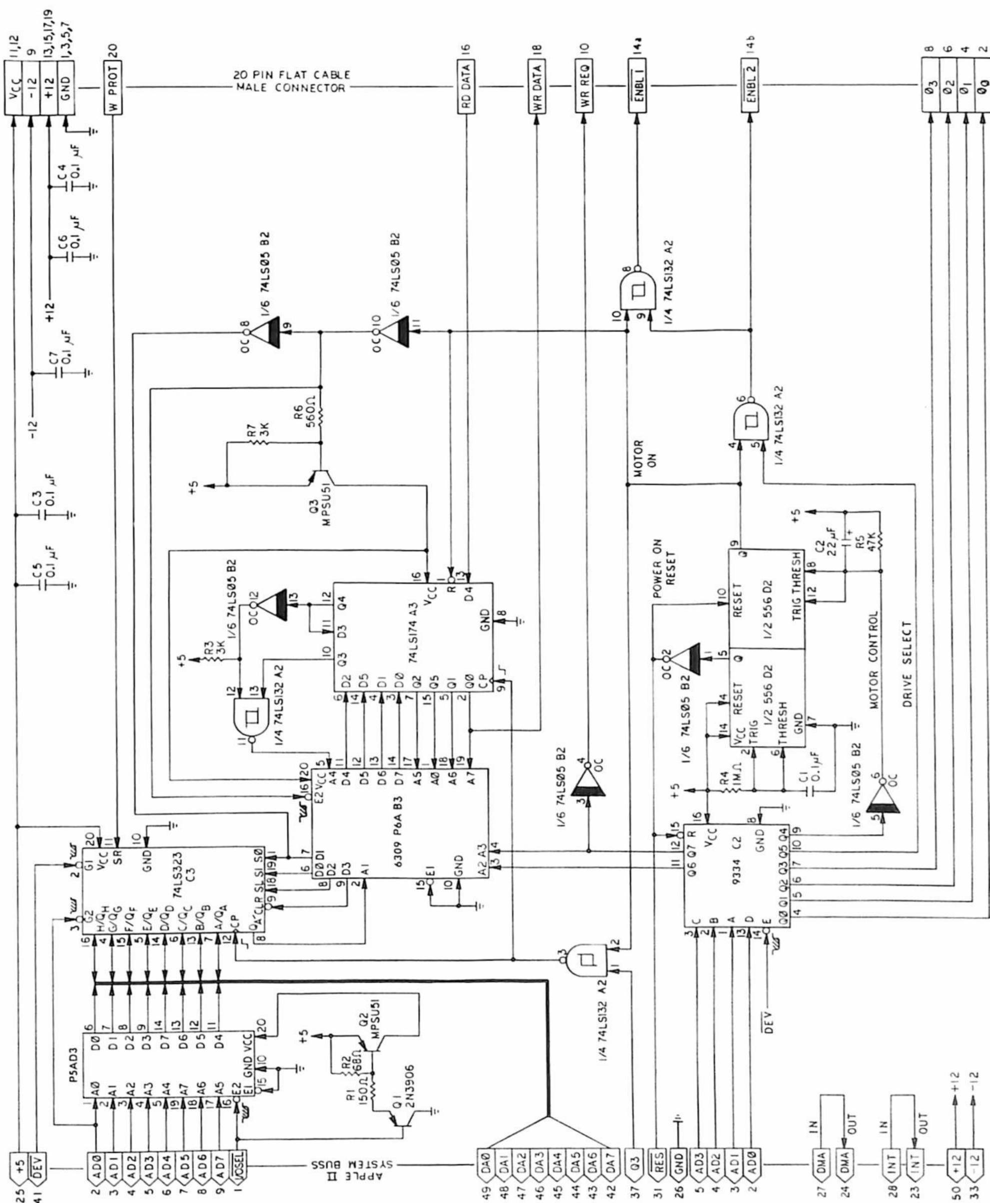
Per osservare l'indirizzo di partenza o la lunghezza dopo un comando BLOAD, battere PRINT PEEK (byte basso)+ PEEK (byte basso+1) *256. Il programma per trovare le posizioni DOS contenenti l'indirizzo di partenza e la lunghezza del programma caricato più recente (BLOAD) su sistemi di qualsiasi dimensione :

```
  5 REM BLOAD FINDER
  7 H=384ØØ : REM DOS-BOOT HIMEM
  8 T=49152 : REM HIGHEST ADDRESS
 1Ø D$=CHR$ (4) REM CTRL-D
 2Ø  PRINT D$; "BSAVE FOO, A$7777, L$77"
 3Ø  PRINT D$; "BLOAD FOO"
 4Ø  PRINT D$; "DELETE FOO"
 5Ø  FOR I = H + 1792 TO T
 6Ø  IF PEEK (I) < > 119 OR PEEK
     (I + 1) < > 119 THEN NEXT 1
 7Ø  PRINT "LOCATIONS OF START"
     ADDRESS : "; I; ", "; I+1
 8Ø  FOR I = H + 1792 TO T
 9Ø  IF PEEK (I) < > 119 OR PEEK
     (I + 1) < > Ø THEN NEXT I
1ØØ  PRINT "LOCATIONS OF LENGTH :"
     ; I; ", ", I + 1
```

I valori di H e T (righe 7 e 8) sono indicati per un sistema di 48K. L'Appendice D, pag. 142, indica i valori correnti per il sistema di cui trattasi. Questo programma richiede all'incirca 2 minuti per trovare le posizioni desiderate.

Routines di input e di output dei caratteri DOS : Vedere Capitolo 10 e particolarmente la Nota 7 a pag. 105. Per un esempio dell'uso della tecnica descritta, vedere il programma a pag. 151.

Schema del circuito : interfaccia DISK II



Schema del circuito : Scheda analogica DISK II



APPENDICE F

Sommario dei comandi DOS

In questa Appendice, i comandi DOS sono raggruppati in cinque categorie:

Comandi di preparazione

INIT	RENAME	VERIFY
CATALOG	DELETE	MON
SAVE	LOCK	NOMON
LOAD		
RUN		

Comandi di accesso

FP	PR#	CHAIN
INT	IN#	

Comandi dei files di testo sequenziali

OPEN	APPEND
CLOSE	POSITION
READ	EXEC
WRITE	

Comandi dei files di testo ad accesso casuale

OPEN	READ
CLOSE	WRITE

Comandi dei files in linguaggio macchina

BSAVE
BLOAD
BRUN

Le procedure usate in DOS (compreso il concatenamento in Applesoft) sono riassunte nell'Appendice G. La notazione usata nei sommari (e nel corso del Manuale) è descritta nel paragrafo che segue.

Notazione

Il termine "sintassi" si riferisce alla struttura di un comando al computer. Viene usata una semplice notazione per descrivere la sintassi di ciascun comando DOS.
Le voci tra parentesi quadre ([e]) sono facoltative. Queste voci sono talvolta dette parametri. Non tutti i comandi ammettono tutti i parametri, ma i parametri ammessi in un dato comando possono comparire in qualsiasi ordine, salvo dove diversamente specificato.
Se un comando usa un nome di file, questo deve venire immediatamente dopo la parola di comando stessa. Il primo elemento che segue il comando sarà trattato come nome file. Il nome file deve essere separato da una virgola da qualsiasi altro parametro che segue.

Le parentesi graffe possono essere usate per indicare quando deve essere premuto un determinato tasto:

– {CTRL} Tenere abbassato il tasto contrassegnato "**CTRL**" mentre viene battuto un altro tasto. {CTRL} D significa tenere abbassato il tasto **CTRL** mentre si batte la lettera D. Viene usata talvolta una altra notazione: **CTRL-D** significa la stessa cosa di {CTRL} D.

– {RETURN} premere il tasto "**RETURN**". Il RETURN richiesto dopo ogni comando non è indicato

– {RESET} premere il tasto contrassegnato "**RESET**"

– {ESC} premere il tasto contrassegnato "**ESC**".

Le lettere maiuscole e le virgole devono essere battute come sono, le lettere minuscole stanno al posto degli elementi che si devono inserire.

f Nome file. Può comprendere da 1 a 30 caratteri. Qualsiasi carattere stampabile salvo la virgola può comparire in un nome file. Il primo carattere deve essere una lettera dell'alfabeto. Per ulteriori particolari, vedere il paragrafo seguente.

Esempio:

CHESS
RECIPE
SUM OF SQUARES
NEW45
HOW-ABOUT-THIS

g Un altro nome file.

Esempio:
SEPARATOR WITH LOW VELOCITY

s Numero di slot. s specifica lo slot di Apple II in cui è stata inserita la scheda unità di controllo del disco (solitamente lo slot 6).
s inizialmente viene presunto allo slot dal quale il DOS è stato lanciato e viene successivamente presunto all'ultimo valore specificato per questo parametro. s deve essere nel campo da 1 a 7.

Esempio:
7/2



Se s si riferisce ad uno slot che non contiene una scheda unità di controllo del disco, il sistema può bloccarsi e il programma nella memoria può anche andar perso. Vedere I/O ERROR, nell'Appendice B, per ulteriori particolari.

v Numero di volume di un mini floppy. v viene inizialmente presunto al numero di volume del disco dal quale il sistema è stato lanciato.
Successivamente viene presunto all'ultimo valore specificato per questo parametro, o implicitamente specificato da un comando CATALOG. v deve essere nel campo da 0 a 254.

Esempio:
1Ø1

Nota: Un numero di volume mini floppy non può essere Ø. In un comando DOS, la specifica di un numero di volume Ø o semplicemente V senza numero dice al DOS di determinare e di usare il numero di volume del disco.

d Numero dell'unità (1 o 2). d è inizialmente presunto a 1 e successivamente viene presunto all'ultimo valore specificato per questo parametro.

Esempio:
2

p Numero di posizione, usato col parametro R nei comandi POSITION ed EXEC per i files di testo sequenziali. p specifica un campo la cui posizione nel file è p campi oltre la posizione corrente nel file. p viene presunto a Ø, per cui non sposta il puntatore della posizione di file. Nota: EXEC definisce sempre il puntatore all'inizio del file indicato, per cui p è sempre relativo a Ø quando usato con EXEC. Vedere i sommari dei comandi più avanti in questa Appendice. p deve essere nel campo da Ø a 32767.

r Numero di record. Usato con il parametro R nei comandi READ e WRITE per i files di testo ad accesso casuale, r è presunto a Ø dopo OPEN. Successivamente, è presunto all'ultimo record specificato, r punta verso un record assoluto all'interno di un file ad accesso casuale. r deve essere nel campo da Ø a 32767.

a Indirizzo in RAM. Il parametro a è richiesto con il comando BSAVE. a specifica l'indirizzo di partenza della memoria di Apple per le informazioni binarie di memorizzazione (BSAVE) o di caricamento (BLOAD). Se BLOAD non specifica un parametro a, il valore di a è presunto pari a quello usato quando il file binario è stato memorizzato (BSAVE). a deve essere nel campo da Ø a 65535.

b Numero del byte. b è presunto a Ø. In un file sequenziale, b punta ad un byte assoluto all'interno del file. In un file ad accesso casuale, b punta ad un byte assoluto all'interno del record puntato da r. b deve essere nel campo da Ø a 32767. Per la maggior parte delle applicazioni, b è nel campo da Ø all'ultimo byte nel file sequenziale corrente o nell'ultimo byte nel record ad accesso casuale corrente.

j Specificatore di lunghezza. j è presunto a 1. Quando usato nel comando OPEN con i files ad accesso casuale, j è obbligatorio e specifica il numero di bytes che costituiranno un record in un file ad accesso casuale. Quando usato con il comando BSAVE, j è obbligatorio e specifica il numero di bytes della memoria di Apple, iniziando con l'indirizzo a, i cui contenuti devono essere caricati su disco. j deve essere nel campo da Ø a 32767.

Come esempio di questa notazione, il comando DOS che è indicato come

INIT f [,Vv] [,Ss] [,Dd]

può essere interpretato come

INIT HELLO, V17, D2

mediante il seguente procedimento. La parola chiave "INIT" è un majuscolo e deve essere battuta esattamente come indicato. Nella descrizione, "f" è in minuscolo e sta per il nome di file – è sostituito dal nome file lecito "HELLO' in questo esempio. Il ",V17" è optional. "V" sta per "volume"; 17 è stato scelto arbitrariamente come numero di volume per questo esempio. La notazione ",Ss" è optional ed è stata omessa. La notazione ",Dd" diventa in questo esempio Dz, indicando che deve essere usata l'unità disco numero 2.

Qualsiasi costante numerica in un comando DOS può essere impostata in notazione esadecimale facendo precedere le cifre esadecimali dal segno del dollaro.

Nomi di file

I nomi di un file possono comprendere fino a 30 caratteri, e devono cominciare con una lettera. Il nome non può contenere una virgola, un **CTRL-M** o un **RETURN**, che è usato per terminare il comando. In un nome, gli spazi che precedono il primo carattere diverso da uno spazio sono ignorati. Tutti i caratteri del nome oltre il trentesimo sono ignorati.



Battendo i nomi di file, l'uso di tasti speciali tipo ad esempio **ESC**, i tasti con la freccia a sinistra o con la freccia a destra e taluni tasti battuti unitamente al tasto **CTRL** (caratteri di "controllo" **CTRL-C, CTRL-H**) possono avere effetti imprevisti.



Se un nome di un file contiene caratteri di controllo, non li vedrete stampati, ma essi devono in ogni caso essere battuti per usare o cancellare il file. Il seguente programma Applesoft può essere usato per trovare qualsiasi carattere nascosto salvo **CTRL-M (RETURN), ESC, CTRL-H** (freccia a sinistra) e **CTRL-U** (freccia a destra).

```
10  DATA 201, 141, 240, 21, 201, 136
20  DATA 240, 17, 201, 128, 144, 13
30  DATA 201, 160, 176, 9, 72, 132
40  DATA 53, 56, 233, 64, 76, 249
50  DATA 253, 76, 240, 253
60  FOR I=768 TO 768+27
70  READ V: POKE I, V: NEXT I
80  POKE 54,0: POKE 55,3
90  CALL 1002
```

Se si pensa di avere accidentalmente introdotto un carattere di controllo in un nome di file, battere questo programma, memorizzarlo (SAVE) ed eseguirlo (**RUN**). Comparirà il segno di richiesta di Applesoft (]). Successivamente battere

CATALOG.

per ottenere un elenco di tutti i files, con qualsiasi carattere di controllo indicato sotto forma di carattere lampeggiante. I caratteri di controllo in un elenco di programma possono a loro volta essere trovati in questo modo. Per ripristinare la normale presentazione, battere

PR#

Comandi di preparazione

INIT f [,Vv] [,Ss] [,Dd]

Esempio:

INIT HELLO, V18

Il parametro v assegna un numero di volume al disco che viene inizializzato.
I particolari sull'inizializzazione dei dischi figurano nel Capitolo 2 e nell'Appendice G.

CATALOG [,Ss] [,Dd]

Esempio:

CATALOG

Presenta sul video il numero di volume ed un elenco di tutti i files sul disco nell'unità specificata o presunta. Se questo comando usa un parametro di volume, [,Vv] quel parametro viene ignorato.

Con ciascun file viene visualizzato un indicatore del tipo di file e il numero dei settori di disco occupati dal file stesso. I tipi di file sono:

- I File di programma Integer BASIC, creato da SAVE
- A File di programma Applesoft BASIC, creato da SAVE
- T File di testo, creato da OPEN e riempito da WRITE
- B File di immagine di memoria binaria creato da BSAVE

Un asterisco a fianco dell'indicatore del tipo di file avverte che quel file è bloccato (LOCK).

Sono disponibili all'utente un massimo di 496 settori di disco, ciascuno dei quali può contenere fino a 256 bytes di informazioni. La lunghezza minima di un file è un settore, per un file di testo vuoto. (Tecnicamente, questo settore è occupato dall'elenco vuoto pista/settore per il file). I files vuoti Integer BASIC, Applesoft e in linguaggio macchina occupano due settori. (Uno per l'elenco pista/settore e uno per il primo settore programma, che contiene la lunghezza del programma. Vedere Appendice C per ulteriori dettagli).



Se un singolo file supera i 255 settori, la presentazione CATALOG della lunghezza di quel file inizia a ØØØ. Ciò non influisce sull'uso del file, ma può dare un'impressione errata di quanto sia stato riempito il disco.

SAVE f [,Ss] [,Dd] [,Vv]

Esempio:

SAVE COLOR DEMOS, V56

Se sul disco nell'unità specificata o presunta, non c'è un file con il nome file specificato, viene creato un file su quel disco e il programma corrente Integer BASIC o Applesoft viene caricato sotto quel dato nome di file. Se il disco contiene un file con il nome di file specificato, ma di un diverso linguaggio o di tipo di file, viene emesso il messaggio

FILE TYPE MISMATCH



Se il disco scelto contiene già un file con il nome file specificato e nello stesso linguaggio, i contenuti del file originale vanno persi e in loro luogo viene memorizzato il programma corrente BASIC. Non viene data alcuna segnalazione.

LOAD f [,Ss] [,Dd] [,Vv]

Esempio:

LOAD DOW JONES, V19, D1

Tenta di trovare sul disco nell'unità specificata o presunta un file di programma Integer BASIC o Applesoft con un nome f. Se i numeri di volume corrispondono e tale file esiste, il programma verrà caricato (LOAD) nel computer. Esso può, quindi, essere elencato (LIST) o eseguito (RUN) o memorizzato (SAVE) come con qualsiasi programma. LOAD chiude qualsiasi file di testo aperto, abilita il linguaggio corretto per il file f, e cancella qualsiasi programma in memoria prima di inserire il nuovo programma in Apple.

Se il file F è un programma Applesoft BASIC e Applesoft non è già in memoria o disponibile dalla scheda ROM in firmware di Applesoft, il programma "Applesoft" verrà automaticamente caricato (LOAD) ed eseguito (**RUN**) dall'unità specificata, prima che il file f sia caricato (LOAD). Se Applesoft non è su quel disco nè sulla scheda ROM in firmware, viene presentato il messaggio.

LANGUAGE NOT AVAILABLE

L'istruzione LOAD, priva di qualsiasi parametro, carica un programma (LOAD) dalla cassetta

RUN f [,Ss] [,Dd] [,Vv]

Esempio:

RUN ANNUITY,D2

Carica (LOAD) il file f dall'unità specificata o presunta (vedere la precedente discussione) quindi, esegue anche il programma caricato (**RUN**). Se viene battuto soltanto

RUN

il programma in memoria viene eseguito (**RUN**)

RENAME f, g [,Ss] [,Dd] [,Vv]

Esempio:

RENAME SEPARATE, SEPARATE, S4, D1, VØ

Trova il file denominato f sul disco nell'unità specificata o presunta e ne cambia il nome in g.I contenuti dei file non sono influenzati. Se il file f era aperto, viene chiuso.



Rename non controlla se il nome file g è già in uso, per cui è possibile usare RENAME per inserire parecchi files con lo stesso nome su un disco – una situazione di potenziale confusione nel migliore dei casi.

DELETE f [,Ss] [,Dd] [,Vv]

Esempio:

DELETE TEST

Rimuove il file f indicato dal disco dell'unità specificata o presunta. Se f era aperto, questo comando lo chiude. Vedere l'Appendice C per ulteriori particolari sul processo di cancellazione.



Se sul disco non esiste un file denominato f compare il messaggio

FILE NOT FOUND

Per evitare che questa circostanza arresti i programmi, aprire per prima cosa il file (OPEN), **quindi** cancellarlo (DELETE).

LOCK f [,Ss] [,Dd] [,Vv]

Esempio:

LOCK LOVE LETTERS, V31

Questo comando consente di proteggere il file f sul disco, nell'unità presunta o specificata, da cancellazioni o cambiamenti accidentali. Un file bloccato (LOCK) è indicato in CATALOG da un asterisco (*).

UNLOCK f [,Ss] [,Dd] [,Vv]

Esempio:

UNLOCK RECIPES, V31, D2

Se si cambia idea e si desidera modificare o rimuovere un file bloccato (LOCK) di nome f, sul disco dell'unità specificata o presunta, questo comando consente tale cambiamento.

VERIFY f [,Ss] [,Dd] [,Vv]

Esempio:

VERIFY SAM

Controlla che l'informazione attualmente caricata sul disco nel file f sia coerente (ecco ciò che succede tecnicamente: quando viene creato un file – con SAVE, BSAVE o WRITE – DOS calcola un byte di somma di controllo per i contenuti di ciascun buffer di output e quindi memorizza quel byte con i contenuti del buffer in un settore del mini floppy. Il comando VERIFY calcola un nuovo byte della somma di controllo per i contenuti effettivi di ciascun settore del file e lo confronta con il byte di somma di controllo originariamente caricato con quel settore). Se i bytes corrispondono (VERIFY) non viene emesso alcun messaggio e si può presumere che le informazioni su quel disco sono state caricate correttamente. In caso contrario, viene presentato il messaggio

I/O ERROR

È possibile verificare (VERIFY) qualsiasi tipo di file.

MON [C] [,I] [,O]

Esempio :

MON O
MON C, I, O

Tutti i comandi del disco e tutte le informazioni scambiate tra computer e disco **non** sono normalmente presentati sul video. Questo comando consente di presentare una parte di o tutte queste informazioni – uno strumento utile quando si esegue il debugging di un programma. Se viene specificato C, vengono presentati i **comandi** del disco. Se viene specificato I vengonno presentate le informazioni inviate dal disco ad Apple, come ad esempio l'**input** di Apple. Se viene specificato O, vengono presentate le informazioni inviate al disco da Apple, come ad esempio l'**output** di Apple.

Deve essere presente almeno uno dei tre parametri, altrimenti MON viene ignorato. I parametri possono comparire in qualsiasi ordine, separati da virgole. Questi parametri compaiono soltanto nei comandi MON e NOMON.

Nota : MON rimane in atto fino a che non incontra un comando NOMON, un cambio di linguaggio (PF o INT), un lancio o una ripartenza (3DØG) del DOS. Anche l'esecuzione di un programma (**RUN**) non cancella un MON.

NOMOM [C], [,I] [,O]

Esempio :

NOMON C
NOMON I, C

Il comando MON consente di presentare i comandi dei dischi e/o informazioni scambiati tra computer e disco : tali informazioni non sono normalmente presentate sul video. Il comando NOMON consente di disabilitare completamente o una parte del video. Il comando

NOMON, C, I, O

riporta il sistema alla sua abituale condizione presunta.

Se viene specificato C, i comandi del disco non vengono presentati. Se viene specificato I, le informazioni invitate dal disco ad Apple, come ad esempio l'input di Apple, non vengono presentate. Se viene specificato O, le informazioni inviate dal disco ad Apple, come ad esempio, l'output di Apple, non vengono presentate.

Deve essere presentato almeno uno dei tre parametri, altrimenti NOMON è ignorato. I parametri possono comparire in qualsiasi ordine, separati da virgole.
Questi parametri compaiono soltanto nei comandi MON e NOMON.

NAXFILES n

Esempio :

MAXFILES 6

n è un numero intero da 1 a 16, che specifica il numero dei files che possono essere attivi contemporaneamente. Quando viene eseguito MAXFILES, per ogni file vengono riservati 595 bytes di memoria (detti buffer dei files). Quando si lancia il sistema, n viene presunto a 3, cosìcchè si hanno 1785 byte riservati per i buffers dei files ed è possibile aprire simultaneamente un massimo di 3 files.

Tutti i comandi DOS salvo PR#, IN# e MAXFILES richiedono un buffer dei files. Pertanto, con MAXFILES e un file aperto (OPEN) un tentativo di eseguire un comando DOS (ad esempio CATALOG) provoca la comparsa del messaggio

NO BUFFERS AVAILABLE



L'uso di MAXFILES sposta HIMEM. Ciò può cancellare i programmi Integer BASIC o le stringhe Applesoft. Usare MAXFILES prima di caricare ed eseguire un programma (LOAD e RUN). Vedere la discussione nei Capitoli 5 e 7 se MAXFILES deve essere usato dall'interno di un programma.

Comandi di accesso

FP [,Ss] [,Dd] [,Vv]

Esempio :

FP, D2

Questo comando mette il sistema in Applesoft BASIC. Qualsiasi programma Integer BASIC o Applesoft in memoria va perso. Se il Computer contiene una scheda Applesoft in firmware, il DOS usa quella sorgente per il linguaggio, indipendentemente dalla posizione dell'interruttore sulla scheda. Se il sistema non contiene la scheda Applesoft in firmware, il DOS tenta di caricare ed eseguire il programma denominato APPLESOFT sul disco nell'unità specificata o presunta.

Per disporre il programma APPLESOFT su un disco di recente inizializzato, caricare innanzitutto il programma APPLESOFT (LOAD) dal disco principale, quindi (senza eseguire – **RUN** – o elencare – LIST – il file), memorizzare APPLESOFT su un disco inizializzato (SAVE). Per questo file occore usare il nome APPLESOFT.



Non usare RUN APPLESOFT per cambiare i linguaggi : dapprima tutto sembra procedere benne, ma il DOS non ha correttamente inizializzato il linguaggio. Per evitare la risultante confusione, usare sempre FP.

INT

Esempio :

INT

Questo comando mette Apple in Integer BASIC. Qualsiasi programma Integer BASIC o APPLESOFT in memoria va perso.

CTRL-D (scritto anche {**CTRL**} **D**)

Esempio :

```
1Ø D$=CHR$(4)
2Ø PRINT D$; "WRITE CHESS"
```

Ogni carattere stampato (PRINT) da Apple viene innanzitutto esaminato dal DOS, prima che venga inviato al mondo esterno. Se Apple stampa (PRINT) un carattere **RETURN** (la maggior parte delle istruzioni PRINT termina automaticamente con un **RETURN**) e il successivo carattere è un **CTRL-D**, questo è un messaggio per il DOS e i successivi caratteri (fino al successivo **RETURN**) sono un comando DOS. La maggior parte dei comandi DOS possono essere usati dall'interno di un programma Integer BASIC o Applesoft. Per fare ciò, creare (PRINT) una stringa composta da **CTRL-D** seguito dal comando DOS desiderato.

Il modo migliore per face ciò consiste nel creare una stringa D$ composta soltanto da un **CTRL-D,** quindi, usare le istruzioni BASIC del tipo indicato nell'esempio. Notare l'uso di CHR$(4) per creare D$ (ciò funziona soltanto in Applesoft, dato che la funzione CHR$ non è presente in Integer BASIC).

Per contro, **CTRL-D** potrebbe essere battuto all'interno di virgolette per creare D$, ma in questo caso non viene presentato alcun carattere tra le virgolette.

Ogni carattere inviato da Apple viene per prima cosa esaminato dal DOS, prima che venga passato al mondo esterno. Se Apple emette un carattere **RETURN** (la maggior parte delle istruzioni PRINT termina automaticamente con un **RETURN**), il successivo carattere è un **CTRL-D**. Si tratta di un segnale per il DOS e i successivi caratteri (fino al successivo **RETURN)** sono un comando DOS. Un comando DOS da un programma deve comparire in un'istruzione PRINT, il cui primo carattere di output è **CTRL-D** e il cui output è separato dal precedente e dal successivo output stampato da veri **RETURN.** Per ulteriori informazioni, vedere "Uso del DOS dall'interno di un programma" nell'Appendice G.

PR#s

Esempio:

PR#6

Invia il successivo output di Apple alla periferica controllata dallo slot #s, invece che al video. Il comando

PR#Ø

invia l'output al video. Se il comando viene usato dall'interno dei programmi, deve comparire come un comando DOS stampato (PRINT) come indicato qui di seguito:

```
1Ø D$="" : REM CTRL-D
2Ø PRINT D$; "PR#1"
```

Se non è installata alcuna scheda unità di controllo di periferica nello slot # s, il sistema può bloccarsi e può dover essere necessario premere il tasto **RESET** per recuperare.

IN#s

Esempio:

IN#6

Prende il successivo input di Apple dalla periferica controllata dallo slot #s, anzichè dalla tastiera di Apple. Il comando

IN#Ø

ripristina il normale input dalla tastiera. Se il comando è usato dall'interno di programmi, deve comparire in un comando DOS stampato (PRINT) come indicato qui di seguito:

```
1Ø D$= "" : REM CTRL-D
2Ø PRINT D$; "IN#1"
```

Se non è installata alcuna scheda unità di controllo di periferica nello slot #s, il sistema può bloccarsi ed occorre premere il tasto **RESET** per ricuperare.

CHAIN f [,Ss] [,Dd] [,Vv]

Esempio:

CHAIN PART TWO, D1, S7, VØ

Usato dall'interno di un programma Integer BASIC, carica ed esegue il programma Integer BASIC di nome f sul disco nell'unità specificata o presunta, ma non cancella i valori di qualsiasi variabile. Ciò significa che il programma f può funzionare sui risultati del programma precedente, e può lasciare i dati per qualsiasi programma seguente. Non è

possibile concatenare (CHAIN) i programmi Applesoft usando questo comando : vedere la procedura speciale per i programmi Applesoft nel Capitolo 10 o nell'Appendice G.

Comandi dei files di testo sequenziale

OPEN f [,Ss] [,Dd] [,Vv]

Esempio :

OPEN SESAMEN D2

Assegna un buffer di memoria di 595 bytes al file di testo f, e prepara il sistema a scrivere o a leggere dall'inizio del file. Questo comando è usato con i comandi WRITE e READ per creare e richiamare files di testo sequenziali.

Se non c'è un file f sul disco nell'unità specificata o presunta, ne viene caricato uno. Se è già aperto (OPEN) un file di nome f, questo comando dapprima lo chiude (CLOSE) prima di aprire (OPEN) in file speficicato.

CLOSE [f]

Esempio :

CLOSE WINDOW

Se si è in scrittura (WRITE), un CLOSE fa sì che tutti i rimanenti caratteri nella parte di output del buffer del file vengano inviati al disco specificato quando quel file è stato aperto (OPEN). CLOSE f disassegna il buffer associato al file di testo sequenziale f. Se CLOSE viene usato senza un nome di file, tutti i files aperti (OPEN) vegono chiusi, ad eccezione del file EXEC. (Può esserci un file EXEC aperto (OPEN) in ogni momento. Quando viene implicitamente aperto un altro file, l'eventuale file esistente EXEC, viene automaticamente chiuso).

Se un programma è interrotto da un **CTRL-C** mentre è aperto un file di testo (OPEN), è una buona idea battere

CLOSE

per evitare che i dati vadano persi.



I files che sono stati assegnati da un'istruzione OPEN devono essere chiusi (CLOSE). La mancata chiusura di un file che era stato aperto e sul quale si era scritto (mediante un comando WRITE) può dar luogo ad una perdita di dati.

WRITE f [,Bb]

Esempio :

WRITE ADRESS.DATA

Dopo questo comando, le istruzioni PRINT inviano il rispettivo output al file specificato, anzichè al video di Apple. Con il parametro Byte, la scrittura (WRITE) inizia al byte b-esimo del file (vedere Capitolo 6, pagina 69). WRITE è cancellato dalla presenza di qualsiasi comando DOS o da un'istruzione INPUT. Basta per questo il comando DOS nullo (ossia la semplice stampa – PRINT – di un **CTRL-D**). WRITE deve essere emesso nel modo ad esecuzione differita.



Dopo questo comando, **tutti** i caratteri di output di Apple che verrebbero normalmente presentati sul video sono invece inviati al disco. Ciò comprende le richieste (punto-interrogativo) di INPUT, i messaggi di errore e qualsiasi altro carattere indesiderato.

READ f [,Bb]

Esempio :

READ SESAME

Dopo questo comando, le istruzioni INPUT, (e le istruzioni GET Applesoft) ottengono i rispettivi caratteri di risposta dal file di testo sequenziale specificato, anzichè dalla tastiera di Apple. Con il parametro Byte, la lettura (READ) inizia al byte b-esimo del file (vedere Capitolo 6, pagina 69).

INPUT fa sì che i caratteri vengano letti (READ) dal file sequenziale un campo alla volta. Un campo si compone di caratteri in numero compreso fra 1 e 32767 e termina con un carattere **RETURN.** In ogni caso, a causa delle limitate capacità delle stringhe dei buffers di input/output, è molto difficile caricare o richiamare campi maggiori di 255 caratteri.

READ è cancellato dalla stampa di qualsiasi comando DOS. Un comando DOS nullo (basta battere – PRINT – un **CTRL-D**) cancella READ. Il comando READ deve essere usato nel modo ad esecuzione differita.

APPEND f [,Ss] [,Dd] [,Vv]

Esempio :

APPEND MORE INFO

Questo comando apre il file di testo specificato ma dispone il puntatore di posizione nel file al **termine** del file stesso. Dopo questo comando, il successivo carattere scritto nel file **seguirà** l'ultimo carattere scritto sequenzialmente e presente nel file. APPEND deve essere seguito da WRITE sul file dello stesso nome. (APPEND non deve essere seguito da OPEN, in quanto OPEN ripristina il puntatore della posizione nel file all'inizio del file).

POSITION f [,Rr]

Esempio :

POSITION ADDRESS.DATA, R277

POSITION dispone il puntatore della posizione nel file all'inizio del campo p-esimo che segue quello in cui attualmente si opera. Un campo è una sequenza di caratteri terminata da un **RETURN.** Le successive letture (READ) o scritture (WRITE) procederanno da quel punto nel file f.

POSITION riguarda una posizione relativa e non assoluta, dato che si contano i campi in avanti da quello in cui ci si trova attualmente nel file, quando viene eseguito POSITION.

POSITION analizza effettivamente in avanti i contenuti del file, carattere per carattere, cercando il carattere **RETURN** p-esimo. Quindi, dispone il puntatore della posizione nel file sul primo byte che segue il carattere **RETURN** p-esimo. Se in questa ricerca trova un qualsiasi byte in cui non è stato memorizzato alcun carattere, fa comparire il messaggio :

END OF DATA

Normalmente, ciò si verifica quando il campo p-esimo davanti alla posizione corrente del file si trova oltre l'ultima entrata del file.

EXEC f [,Rr] [,Ss] ([,Dd] [,Vv]

Esempio :

EXEC UTILITY

Simile a **RUN**, salvo che f è un file di testo (dati) contenente comandi BASIC e DOS così come verrebbero inseriti dalla tastiera. Ciò consente di predisporre i files che possono controllare Apple, più o meno come se l'utente controllasse da sè Apple.

Può esserci soltanto un comando EXEC in atto in ogni momento. Se il file EXEC contiene il comando ad esecuzione immediata EXEC, il file EXEC originale viene chiuso e viene aperto ed eseguito il nuovo file EXEC. Se EXEc ha aperto (OPEN) un file, il comando

CLOSE

inon chiude (CLOSE) il file in corso di esecuzione (EXEC). Quando un file EXEC ha completato tutti i suoi comandi, si chiude (CLOSE) e si ferma. Se un file in corso di esecuzione (EXEC) contiene un comando per eseguire (**RUN**) un qualsiasi programma, EXEC attende pazientemente che il programma termini, quindi, viene eseguito il successivo comando nel file EXEC.



Per contro, se un programma è in fase di esecuzione mentre è aperto un file EXEC, qualsiasi istruzione INPUT nel programma prende come risposta il successivo campo del file EXEC, ignorando la tastiera. Peggio ancora, se quella risposta è un comando DOS ad esecuzione immediata, il comando viene **eseguito** prima che il programma continui.



Se si batte **CTRL-C** per interrompere un programma Applesoft che è in fase di esecuzione mentre è ancora aperto un file EXEC, i rimanenti comandi nel file EXEC **non vengono** solitamente eseguiti.

Se si specifica il valore del parametro R, viene posto un puntatore della posizione nel file all'inizio del campo p-esimo del file, e EXEC inizia l'esecuzione da questo punto nel file.

Come con POSITION, il parametro R usato con EXEC va inteso come parametro di posizione di campo relativa. Per contro, a differenza di POSITION, EXEC conta **sempre** i campi **dal l'inizio** del file, per cui p è sempre relativo a Ø. Gli altri parametri funzionano come al solito.

Se si specifica il valore del parametro R oltre il termine del file compare un messaggio

END OF DATA

Comandi dei files di testo ad accesso casuale

OPEN f, Lj [,Ss] [,Dd] [,Vv]

Esempio :

OPEN SESAME, L2

OPEN assegna un buffer dei files di 595 bytes al file di testo ad accesso casuale f, e definisce la lunghezza el record al numero di bytes specificato da j. Il numero j deve essere nel campo da 1 a 32767 ed è presunto a 1.

OPEN viene usato con i comandi read e WRITE per creare e richiamare files di testo ad accesso casuale. Notare che il parametro L (lunghezza) non è facoltativo : per definizione occorre specificare la lunghezza del record di un file di testo ad accesso casuale. Ogniqualvolta si usa un particolare file di testo ad accesso casuale, occorre aprire (OPEN) il file con lo stesso valore di parametro L. DOS usa quindi quel valore per calcolare la posizione di inizio di qualsiasi record specificato.

Se non c'è un file f, ne viene creato uno.

CLOSE [f]

Esempio :

CLOSE BOOK

Se si sta scrivendo (WRITE), un CLOSE fa sì che tutti i rimanenti caratteri nella parte di output del buffer dei files vengano inviati al disco dell'unità specificata nel momento in cui il file è stato aperto (OPEN), CLOSE disassegna il buffer associato al file di testo ad accesso casuale f. Se CLOSE viene usato senza un nome di file, tutti i files OPEN saranno chiusi, ad eccezione del file EXEC se presente.

Se un programma viene interrotto da un **CTRL-C** mentre un file di testo è OPEN, è una buona idea battere

CLOSE

per evitare di perdere i dati.



I files che erano stati assegnati da un'istruzione OPEN devono essere chiusi (CLOSE). La mancata chiusura di un file che era stato aperto e sul quale si era scritto (con un WRITE), può dar luogo a perdita di dati.

WRITE f [,Rr] [,Bb]

Esempio :

WRITE ADDRESS.DATA, R3

Dopo questa istruzione, le istruzioni PRINT inviano il loro output al file specificato, anzichè al video di Apple. WRITE è cancellato dalla presenza di un qualsiasi comando DOS o da un comando INPUT. Il comando DOS nullo (la semplice battuta di un **CTRL-D**) interrompe un WRITE. WRITE può essere usato soltanto nelle istruzioni PRINT ad esecuzione differita.

Il parametro R (Record) fa sì che WRITE inizi al primo byte del record r-esimo, dove ciascun record contiene il numero di bytes j specificato dal parametro L dato con OPEN. r è presunto a Ø. Se viene specificato il parametro B, WRITE inizia al byte b-esimo del record r-esimo nel file.



Dopo l'istruzione WRITE, **tutti** i caratteri di output di Apple che verrebbero normalmente presentati sul video sono, invece, inviati al disco. Ciò comprende le richieste di INPUT sotto forma di punto di domanda, i messaggi di errore e altri caratteri indesiderati.

READ f [,Rr] [,Bb]

Esempio :

READ SESAME,R3,B3Ø

Dopo questa istruzione, le istruzioni INPUT (e le istruzioni GET in Applesoft) ottengono i rispettivi caratteri di risposta dal file di testo ad accesso casuale specificato, anzichè dalla tastiera di Apple. INPUT fa sì che i caratteri vengano letti (READ) dal record corrente del file ad accesso casuale, un campo alla volta.

Un campo può avere da 1 a 32767 caratteri, e terminare con un carattere **RETURN**. In ogni caso, nessun record deve essere lungo più di j caratteri, dove j è la lunghezza del record specificata quando il file è stato aperto.

Il parametro R (Record) fa sì che la lettura (READ) inizi al primo byte del record r-esimo, dove ciascun record contiene il numero di bytes j, specificato dal parametro L, dato con OPEN. r è presunto a Ø. Se è specificato il parametro B, il READ inizia al byte b-esimo del record r-esimo nel file.

READ è cancellato dalla presenza di un qualsiasi comando DOS. Un comando DOS nullo (la semplice battuta di **CTRL-D**) cancella READ.

Comandi dei files in linguaggio macchina

BSAVE f, Aa, Lj [,Ss] [,Dd] [,Vv]

Esempi :

BSAVE PICTURE, A16384, L8192
BSAVE PICTURE, A$4ØØØ, L$2ØØØ

Crea un file di nome f e memorizza i contenuti di un segmento della memoria di Apple II. Il segmento è specificato dall'indirizzo di partenza a, e dal numero di bytes da caricare j.

Gli esempi sopra indicati memorizzano un'immagine ad alta risolvenza, dalla seconda area di immagini ad alta risolvenza. Essi sono operativamente identici : il secondo esempio semplicemente usa la notazione esadecimale per i parametri.

BLOAD f [,Aa] [,Ss] [,Dd] [,Vv]

Esempi :

BLOAD PICTURE, A8192
BLOAD PICTURE, A$2ØØØ

Se a non è specificato, BLOAD inserisce il file specificato nella memoria di Apple, iniziando dalla posizione di partenza dell'area di memoria che era stata originariamente caricata (BSAVE). Se a è specificato, i dati vengono inseriti nella memoria di Apple a partire dall'indirizzo a. Notare che un programma in linguaggio macchina potrebbe non essere più eseguibile se viene così spostato.

Si supponga che su un disco sia stata memorizzata (BSAVE) un'immagine di grafici ad alta risolvenza sotto il nome di file PICTURE. Il primo esempio sopra indicato disporrebbe, quindi, le immagini nella prima area di immagini ad alta risolvenza, che inizia alla posizione di memoria 8192 (decimale). Il secondo esempio è equivalente : l'indirizzo è indicato in esadecimale, come denota la presenza del segno del dollaro prima di 2ØØØ.



Entrambi gli esempi rovinerebbero qualsiasi versione di Applesoft che non fosse in firmware (ROM).

BRUN f [,Aa] [,Ss] [,Dd] [,Vv]

Esempio :

BRUN SUPER, A$CØA, V75

Carica (BLOAD) il file f nella memoria di Apple, a partire dalla posizione a. Se il parametro a viene omesso, il file viene caricato (BLOAD) a partire dalla stessa posizione dalla quale era stato memorizzato BSAVE. Una volta caricato (BLOAD) il file (che deve essere un programma in linguaggio macchina) viene iniziato da un salto in linguaggio di macchina (JMP) alla posizione a.

APPENDICE G

Sommario delle procedure DOS

Questa appendice contiene i sommari delle principali procedure usate in DOS. Nella pagina precedente, queste sono elencate con i numeri di pagina in cui esse compaiono.

Lancio del DOS (Booting DOS)

Sostituire "s" con il numero di slot in cui è posta l'unità di controllo del disco.

Carattere di richiesta	Linguaggio	Per lanciare il DOS, battere
>	Integer BASIC	PR#s oppure IN#s
]	Applesoft	PR#s oppure IN#s
*	Monitor	s {CTRL}P

Inizializzazione di un mini floppy

Per inizializzare (INIT) un disco secondario (dipendente dalla memoria) :
1) Lanciare il DOS
2) Inserire un disco vuoto nell'unità disco
3) Battere un programma "greeting", ad esempio.

```
1Ø PRINT "32K SLAVE DISKETTE INITIALIZED 5 MAY 8Ø"
2Ø END
```

4) Supponendo di aver scelto il nome "HELLO" per il programma "greeting", battere il comando :

```
INIT HELLO
```

5) Dopo lo spegnimento della spia IN USE sull'unità disco, rimuovere il disco e contrassegnarlo.

Per creare un disco principale (indipendente dalla memoria) del programma MASTER CREATE.

Recupero da errori accidentali

Se DOS è stato lanciato e quindi è stato accidentalmente premuto il tasto **RESET**, battere CALL – 151 **RETURN**.

```
3DØG RETURN
```

(cioè il numero zero dopo la D) per ritornare al BASIC precedentemente lasciato, con il programma intatto.

Uso del DOS all'interno di un programma

I comandi DOS possono essere emessi dall'interno di un programma battendo (PRINT) **CTRL-D** e quindi il comando. Creare dapprima una stringa D$ che si compone soltanto di **CTRL-D**.
In Applesoft, D$ può essere creato dal comando

```
D$=CHR$(4)
```

dato che **CTRL-D** è il carattere il cui codice ASCII è 4.
In entrambi i BASIC, D$ può essere definito battendo D$="
quindi tenendo abbassato il tasto **CTRL,** mentre si battre la lettera D e quindi battendo le virgolette di chiusura. I caratteri di controllo tipo **CTRL-D** non sono presentati, cosìcchè ciò che si vede è D$=" "

Questo programma Applesoft presenta il CATALOG quando viene eseguito **RUN** :

```
5  REM GREETING PROGRAM
1Ø D$= CHR$ (4) : REM CTRL-D
2Ø PRINT D$; "CATALOG"
```

In una singola istruzione PRINT può essere contenuto soltanto un comando DOS. I contenuti fra virgolette dell'istruzione PRINT devono cominciare con un CTRL-D e terminare con il comando DOS. CTRL-D deve essere preceduto da RETURN (inviato automaticamente al termine della istruzione PRINT).

Questi comandi devono essere usati soltanto nel modo ad esecuzione differita (dall'interno di un programma), comparendo dopo CTRL-D in un'istruzione PRINT :

```
OPEN APPEND READ WRITE POSITION
```

Altri comandi diversi possono essere usati sia nel modo ad esecuzione immediata che dall'interno di un programma dove essi compaiono dopo un **CTRL-D** in un'istruzione PRINT.

Creazione di un sistema automatico (Chiavi in mano)

Per preparare un disco che esegua un certo programma ogniqualvolta il disco viene lanciato – nell'esempio useremo il programma COLOR DEMO – adottare la seguente procedura :

1. Inizializzare un disco vuoto (INI) usando il nome HELLO per il programma "greeting".
2. Disporre un disco contenente il programma COLOR DEMOS nell'unità, e battere :

```
RUN COLOR DEMOS
```

Una volta sicuri che il programma funziona (RUN) correttamente, ritornare al BASIC.

3. Inserire il disco di recente inizializzato nell'unità disco e battere:

SAVE HELLO

per sostituire il vecchio programma "greeting" col programma COLOR DEMOS.

Creazione e richiamo di files di testo sequenziali

Creando un file di testo sequenziale, un OPEN deve precedere un WRITE; una volta che il WRITE viene eseguito, qualsiasi successivo comando PRINT invia tutti i caratteri al disco.
Chiudere (CLOSE) il file una volta terminato. Un comando WRITE è cancellato da un INPUT oppure dall'uso di un qualsiasi comando DOS in un'istruzione PRINT (basta anche **CTRL-D**).

Questo programma Applesoft crea un file di testo sequenziale di nome SAMPLE i cui primi 13 campi contengono tre stringhe e gli interi da 1 a 1Ø:

```
  5 REM MAKE SAMPLE
 1Ø D$=CHR$ (4) : REM CTRL-D
 2Ø  PRINT D$; "OPEN SAMPLE"
 3Ø  PRINT D$; "DELETE SAMPLE"
 4Ø  PRINT D$; "OPEN SAMPLE"
 5Ø  PRINT D$; "WRITE SAMPLE"
 6Ø  PRINT "HI HO" : PRINT "HI HO"
 7Ø  PRINT "OFF TO THE DISK WE GO"
 8Ø  FOR J=1 TO 1Ø
 9Ø  PRINT J : NEXT J
11Ø  PRINT D$; "CLOSE SAMPLE"
```



Se si apre un file che già esiste e quindi vi si scrive (WRITE), si cancella una parte del file originale. Questo programma Applesoft richiama il file SAMPLE sopra descritto, un campo alla volta. Se si desidera vedere cosa viene letto (READ) dal disco, il comando

MON I

fa comparire l'input dal disco.

```
  5 REM RETRIEVE SAMPLE
 1Ø D$=CHR$ (4) : REM (CHR$(4)
      IS CTRL-D
 2Ø  PRINT D$; "OPEN SAMPLE"
 3Ø  PRINT D$; "READ SAMPLE"
 4Ø  INPUT A$, B$, C$
 5Ø  FOR I=1 TO 1Ø
 6Ø  INPUT W
 7Ø  NEXT I
 8Ø  PRINT D$; "CLOSE SAMPLE"
```

Un OPEN deve precedere un READ. Una volta eseguito READ, qualsiasi successiva istruzione INPUT (in Applesoft, anche qualsiasi istruzione GET), ottiene i rispettivi caratteri di risposta dal disco, anzichè dalla tastiera di Apple. Chiudere (CLOSE) il file al termine.

Un READ viene cancellato battendo **CTRL-D**, seguito o meno da un comando DOS.

Aggiunta di dati ad un file di testo sequenziale

Questo comando Applesoft aggiunge le due stringhe "TEST 1" e "AND NOW FOR TEST 2" al termine del file di testo sequenziale di nome SAMPLE. Ciascuna stringa è un campo supplementare del file.

```
 5 REM APPEND SAMPLE
10 D$= CHR$ (4) : REM CTRL-D
20  PRINT D$; "APPEND SAMPLE"
30  PRINT D$; "WRITE SAMPLE"
40  PRINT "TEST 1"
50  PRINT "AND NOW FOR TEST 2"
60  PRINT D$; "CLOSE SAMPLE"
```

Controllo di apple mediante un file di testo sequenziale

Quando eseguito (RUN), questo programma Applesoft crea un file di testo di nome DOIT, contenente i comandi

```
LIST 20, 50
RUN HELLO
CATALOG
```

```
 5 REM MAKE DOIT
10 D$= CHR$ (4) : REM CTRL-D
20  PRINT D$; "OPEN DOIT"
30  PRINT D$; "WRITE DOIT"
40  PRINT "LIST 20, 50"
50  PRINT "RUN HELLO"
60  PRINT "CATALOG"
70  PRINT D$; "CLOSE DOIT"
```

Una volta creato il file di testo DOIT, il comando

```
EXEC DOIT
```

fa sì che i comandi nel file DOIT vengano eseguiti uno alla volta, esattamente come se fossero battuti dalla tastiera.

Creazione e richiamo di files di testo ad accesso casuale

Questo programma Applesoft crea un file di testo ad accesso casuale di nome RA-FILE, i cui records sono lunghi ciascuno 30 bytes, quindi scrive la stringa "NAME ADDRESS" seguita dal numero di record nei record da 12 a 15 del file. Nelle righe 7Ø e 8Ø, il numero di record 13 è cambiato per accogliere la stringa "DOS".

```
  5 REM MAKE RA-FILE
 1Ø D$=CHR$ (4) : REM CTRL-D
 2Ø  PRINT D$; "OPEN RA-FILE"
 3Ø  PRINT D$; "DELETE RA-FILE"
 4Ø  PRINT D$; "OPEN RA-FILE, L3Ø"
 5Ø  FOR I=12 TO 15
 6Ø  PRINT D$; "WRITE RA-FILE, R"; I
 7Ø  PRINT "NAME ADDRESS"; I
 8Ø  NEXT 1
 9Ø  PRINT D$; "WRITE RA-FILE, R13"
1ØØ  PRINT "DOS"
11Ø  PRINT D$; "CLOSE RA-FILE"
```

Questo programma Applesoft legge (READ) i records da 12 a 15 del file di testo ad accesso casuale RA-FILE. Notare che occorre specificare ciascun record prima di leggerlo (READ) nella riga 4Ø. La riga 6Ø esamina i tre caratteri più a sinistra della stringa di input A$, prelevati da ciascun record. Se questi tre caratteri sono "DOS" viene battuto il messaggio "RECORD r WAS CHANGED", e la ricerca continua.

```
 5 REM RETRIEVE RA-FILE
1Ø D$= CHR$ (4) : REM CTRL-D
2Ø  PRINT D$; "OPEN RA-FILE, L3Ø"
3Ø  FOR J=12 TO 15
4Ø  PRINT D$; "READ RA-FILE, R"; J
5Ø  INPUT A$
6Ø  IF LEFT$ (A$, 3)= "DOS" THEN
    PRINT "RECORD"; J; WAS CHAN
     GED. "
7Ø  NEXT J
8Ø  PRINT D$; "CLOSE RA-FILE"
```

Copiatura di un file di testo

La copia di un programma BASIC o di un File Binario, su un altro mini floppy, non è un problema : basta caricare (LOAD o BLOAD) i contenuti dei files nell'Apple, e quindi salvarli su un altro mini floppy (SAVE o BSAVE).

È necessario specificare l'indirizzo e la lunghezza nell'uso del comando BSAVE.

Il programma COPY permette di generare copie complete di mini floppy, ma se qualche File è protetto da copiatura, otterete il messaggio di errore :

```
I/O ERROR
STOPPED AT 32Ø
```

Per copiare un file di testo, o qualsiasi file non protetto, usare il programma FID, le cui istruzioni si possono trovare nell'Appendice J.

Concatenamento in applesoft

Per eseguire (RUN) una serie di programmi Applesoft senza cancellare i precedenti valori delle variabili e delle matrici usare la seguente procedura.

Supponendo di voler concatenare il programma PARTE DUE al programma PARTE UNO, essere sicuri che il File Binario CHAIN sia presente sullo stesso disco che contiene il programma PARTE DUE (se non c'è, usare il programma FID per ottenerne una copia).

Quindi inserire semplicemente le seguenti due linee BASIC nel programma PARTE UNO, quali ultime linee da eseguire prima di passare il controllo al programma PARTE DUE :

```
PRINT CHR$(4); "BLOAD CHAIN,A52Ø"
CALL 52Ø"PARTE DUE"
```

Nella istruzione CALL non sono ammessi spazi o altri caratteri fra lo zero (Ø) e il doppio apice (").

APPENDICE H

Aggiornamento del DOS per ottenere 16 settori

Se esiste già un Disk Operating System e se viene usata una versione del DOS che gira su 13 settori (DOS 3.2.1. o precedenti), occorre cambiare 2 PROM sulla scheda del controller per poter ottenere un sistema a 16 settori. Qualsiasi versione del DOS precedente alla release 3.3 dovrà essere aggiornata.

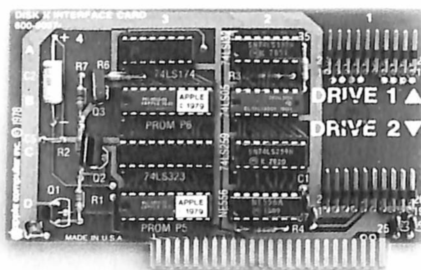
Come installare le nuove PROMS

Le nuove PROM dell'unità floppy disk sostituiscono le 2 PROM su ciascuna scheda unità di controllo Disk II di Apple.

All'interno dell'apparecchio, individuare gli "slots" connettore laterale nella parte posteriore della scheda principale. In uno o più slots si troveranno le schede contrassegnate "Disk II Interface Card" nel bordo superiore. Assicurarsi di aver spento l'apparecchio. Rimuovere ciascuna di queste schede dell'unità di controllo afferrandole sul bordo superiore e facendole oscillare delicamente avanti ed indietro. Rimuovere i cavi a nastro da ciascuna scheda, in modo da poter lavorare meglio.

Afferrare una scheda unità di controllo con il lato che porta i circuiti integrati in avanti e la parte a pettine verso il basso. Sulla scheda si potranno vedere 4 file di circuiti integrati. Individuare il circuito integrato di sinistra nella seconda fila a partire dall'alto, che dovrebbe essere contrassegnato con "P6" oppure con "31-ØØ1Ø-xx" (eventualmente sotto un adesivo di copy-right) mentre la scheda sottostante dovrebbe essere contrassegnata "PROM P6". Questo è il circuito integrato che occorrerà sostituire con la nuova PROM P6 A.

Individuare ora il circuito integrato di sinistra nella fila inferiore. Questo dovrebbe essere contrassegnato "P5" oppure "341-ØØØ9-xx" (eventualmente sotto un adesivo di copyright) e la scheda sottostante dovrebbe essere contrassegnata "PROM P5". Questo è il circuito integrato che occorre sostituire con la nuova PROM P5A.



Osservare ora le nuove PROM. La nuova PROM P6A dovrebbe essere contrassegnata con "P6A" o con "341-ØØ28-xx" o con entrambe le scritte. La nuova PROM P5A dovrebbe essere contrassegnata con "P5A" oppure con "341-ØØ27-xx" o con entrambe le scritte. Dovrebbe essere possibile trovare questi riferimenti senza togliere gli adesivi. Se non è possibile trovare questi numeri, consultare il rivenditore.

Ora che si conosce la posizione di tutto ciò che serve, si possono sostituire le vecchie PROM con le nuove. Disporre la scheda unità di controllo su un tavolo e tenerla saldamente. Servendosi di un estrattore per circuiti integrati, rimuovere delicatamente la PROM P6 dal suo zoccolo. Sulla scheda si può trovare un transistor, contrassegnato "Q3" che blocca l'estrattore del circuito integrato. In questo caso, piegare delicatamente spostandolo lateralmente fino a che non è quasi verticale. Afferrare il circuito integrato con l'estrattore e farlo oscillare delicatamente per estrarlo dallo zoccolo.

Tenere la PROM P6A, contrassegnata con "P6A" o con "341-ØØ28-xx" in modo che sia orientata allo stesso modo dei circuiti integrati della scheda. La grande tacca rettangolare dovrebbe trovarsi sul lato sinistro guardando la scheda (potrebbe esservi una piccola depressione circolare sull'altra estremità. Se non si è sicuri, chiedere al rivenditore).

Assicurandosi che ciascun pin si allinei con il relativo foro, inserire delicatamente la PROM P6A nello zoccolo P6. Se i pins sono troppo deformati per poterli introdurre facilmente, raddrizzarli fino a che non sono verticali premendoli delicatamente (molto delicatamente) contro la superficie di un tavolo.

Rimuovere ora la PROM P5 e sostituirla con la PROM P5A, contrassegnata con "P5A" oppure con "341-Ø027-xx" esattamente come è stato fatto con la P6A che ha sostituito la P6.

Se ci sono più di 2 unità floppy Disk II, occorre una scheda unità di controllo per ciascuna delle due unità. Ciascuna di queste schede avrà bisogno delle nuove PROM. Installarle come già fatto per la prima coppia.

Unità floppy disk multiple

Collegare ora ciascuna unità floppy Disk II alla relativa unità di controllo ed inserire la scheda nel suo slot, nell'ordine indicato nella tabella che segue. La prima unità dovrebbe andare nella prima posizione indicata (slot 6, unità 1) e così via fino a che non sono state installate tutte le unità. Si procederà verso il basso a partire dallo slot 6, occupando per primi i connettori drive 1 e quindi drive 2 di ciascuna scheda. Ciò assicura che le unità siano nelle posizioni più efficienti, aumentando l'affidabilità del sistema: i programmi non tenteranno cioè di usare unità inesistenti o di ignorare le unità esistenti.

POSIZIONAMENTO DELLE UNITÀ DISCO DISK II

Ordine di installazione	Numero slot	Numero unità
1a unità	6	1
2a unità	6	2
3a unità	5	1
4a unità	5	2
5a unità	4	1
6a unità	4	2

Contrassegnare ciascuna unità "DRIVE 1" o "DRIVE 2" a secondo dei casi.

Assicurarsi che tutti i collegamenti siano correttamente eseguiti e che tutto sia stato installato nella corretta posizione. La PROM P5 di ciascuna scheda unità di controllo dovrebbe essere stata sostituita dalla PROM P5A (341-ØØ27-xx). La PROM P6 di ciascuna scheda unità di controllo dovrebbe essere stata sostituita dalla PROM P6A (341-ØØ28-xx). Nel connettore drive 1 di ciascuna scheda unità di controllo Disk II dovrebbe essere inserito un cavo a nastro Disk II. Lo slot 6 dovrebbe avere avere una unità di controllo.

Quando tutto è installato correttamente, rimontare il coperchio di Apple e procedere all'aggiornamento degli esistenti mini floppy a 13 settori per portarli a 16 settori, usando il programma MUFFIN.

Uso del DOS con la language system

È possibile usare il DOS sia con Applesoft che con Integer BASIC e con l'Apple Language System installato. Per far ciò, non occorre il diskette BASICS. Per iniziare un DOS a 16 settori dopo aver usato qualsiasi linguaggio sul Language System che usa il Pascal Operating System, seguire questa procedura:

1. Ricordarsi di memorizzare qualsiasi programma sul quale si sta lavorando, altrimenti, iniziando il DOS, questi verranno cancellati dalla memoria.

2. Entrare nel modo Command del Pascal Operating System (in modo da poter vedere la riga che inizia con "Command:").

3. Sostituire il mini floppy nel Drive 1 (collegato alla scheda da unità di controllo nello slot di numero più elevato) con il diskette DOS System Master.

4. Battere H (per Halt). Ciò fa sì che l'unità disco lanci il diskette System Master.

Il mini floppy System Master caricherà ora Applesoft o Integer BASIC – quello dei due che non è già disponibile sulla scheda principale – nello spazio di memoria disponibile sulla Language Card, una volta lanciato, caricherà Applesoft nella Language Card. Se si dispone di un Apple II Plus, verrà caricato Integer BASIC. In entrambi i casi, comparirà un messaggio, indicante quale BASIC ha caricato durante il lancio del System Master. Tutti i comandi DOS funzioneranno per usare e scrivere programmi in Applesoft e Integer Basic.

Per imparare la procedura per iniziare il DOS dopo l'uso di un linguaggio che non usa il Pascal Operating System, vedere il manuale per quel linguaggio.

Con il Language System installato, è possibile lanciare il DOS all'accensione di Apple, col mini floppy System Master nel Drive 1 o unità disco 1.

Per ritornare dal DOS ad un altro linguaggio Language System, sostituire il System Master nel Drive 1 con il mini floppy di lancio per quel linguaggio. Quindi battere PR#6 e premere **RETURN**.

Come personalizzare il lancio procedura con language card

Quando si inserisce un mini floppy DOS 3.3 nel drive 1 e si accende l'apparecchio, Apple esegue il programma HELLO scritto su quel mini floppy, purchè contenga nella ROM il BASIC in cui è scritto il programma HELLO. Apple II eseguirà cioè un programma HELLO Integer BASIC, Apple II Plus eseguirà un programma Applesoft HELLO e Apple II con la scheda Applesoft o Apple II Plus con la scheda Integer BASIC eseguirà un programma HELLO con l'uno e l'altro BASIC. Se non c'è quel BASIC nella ROM, si ottiene un messaggio di errore:

LANGUAGE NOT AVAILABLE

e la richiesta per il BASIC mancante.

Se si dispone di un Apple Language System, le cose si fanno un pochino più complicate. Senza ulteriori problemi, è possibile eseguire programmi nel BASIC esistente nella ROM, ma prima di poter usare l'altro BASIC, occorre caricarlo nella Language Card. Il modo più semplice per far ciò consiste nel lanciare il mini floppy System Master DOS 3.3 che è fornito con due programmi, HELLO e APPLESOFT, che insieme assicurano che entrambi i BASIC siano caricati quando si lancia il DOS. HELLO, scritto in Applesoft per Apple II plus, carica Integer BASIC. Applesoft, scritto in Integer BASIC per Apple II, carica Applesoft. La macchina eseguirà l'appropriato programma e successivamente si comporterà come se nella ROM disponesse di entrambi i BASIC. È ora possibile inserire qualsiasi altro mini floppy DOS 3.3 ed eseguire su di esso qualsiasi programma.

Se si vuole preparare un mini floppy che funzioni su tutti i sistemi, e carichi la Language Card, se necessario, è possibile farlo copiando su di esso gli appropriati files. Ecco come:

1. Lanciare il mini floppy System Master DOS 3.3

2. Caricare HELLO dal System Master, quindi inserire un mini floppy vergine e battere

   INIT HELLO

3. Quando INIT è terminato, reinserire il System Master, BRUN MASTER CREATE e battere

   HELLO

   quando viene richiesto il nome del programma di saluti. Inserire il mini floppy vergine quando MASTER CREATE dice di farlo.

4. Usando FID, copiare questi files sul mini floppy di recente inizializzato:

   APPLESOFT
   INTBASIC
   FPBASIC

5. Inserire qualsiasi programma ed altri files a piacere su questo mini floppy. Assicurarsi di avere almeno i seguenti files:

   HELLO
   APPLESOFT
   INTBASIC
   FPBASIC

Quando si lancia questo mini floppy su un Apple con la Language Card, questo caricherà la Language Card, con il BASIC non eistente nella ROM, cosicchè Apple può eseguire qualsiasi programma nell'uno o nell'altro BASIC.

Se si vuole avere un programma che Apple esegua automaticamente quando viene acceso, si possono modificare i programmi HELLO e APPLESOFT In modo che ciascuno di essi esegua il programma scelto (che chiameremo qui TURNKEY). Si troverà più facile eseguire l'edit di questi programmi se dapprima si esegue RUN LOADAPA, quindi si usa il comando &S, in modo da poter avere il **CTRL-D** in ciascun comando DOS incorporato.

Iniziare con il mini floppy appena preparato nelle fasi da 1 a 5, e procedere come segue:

6. UNLOCK HELLO
7. LOAD HELLO

8. Elencare (LIST) ed osservare le ultime righe

```
240  END
250  REM
260  REM -- NO CARD OR CAN'T RELOAD
270  REM
280  IF PEEK (768)=0 THEN END
290  PRINT: PRINT "...LANGUAGE CARD CANNOT BE RELOADED":
     PRINT" UNTIL THE SYSTEM IS REBOOTED..."
300  END
```

Sul video, le righe saranno limitate a 40 caratteri; qui sono suddivise in altro modo per maggior chiarezza.

9. Cambiare queste righe come segue:

```
*240  GOTO 300
 250  REM
 260  REM -- NO CARD OR CAN'T RELOAD
 270  REM
*280  IF PEEK (768)=0 THEN 300
 290  PRINT: PRINT "...LANGUAGE CARD CANNOT DE RELOADED";
      PRINT" UNTIL THE SYSTEM IS REBOOTED..."
*300  PRINT "ctrl-dRUN TURNKEY"
```

Le righe cambiate o aggiunte sono contrassegnate con asterischi. Il ctrl-d nella riga 300 rappresenta un CTRL-D che non comparirebbe sul video o in un elenco. Se non si desiderano nei files i caratteri di controllo invisibili è possibile dichiarare una stringa D$ contenente un CTRL-D, che farebbe apparire così la riga 310 (in Applesoft):

```
*310 D$=CHR$(4): PRINT D$:"RUN TURNKEY"
```

Così facendo, assicurarsi di dichiarare D$ prima di usarlo.

È possibile anche cambiare o cancellare alcune istruzioni PRINT nel programma, ma in questo caso bisogna assicurarsi di lasciare quelle con i comandi DOS (cioè quelle che hanno i CTRL-D)

10. SAVE HELLO

11. LOCK HELLO

12. Sbloccare (UNLOCK) APPLESOFT, caricarlo (LOAD) ed elencarlo (LIST) osservando le ultime righe:

```
250  GOTO 310
260  REM
270  REM -- NO CARD OR CAN'T RELOAD
280  REM
290  IF PEEK ()(+)=0 THEN 310
300  PRINT "...LANGUAGE CARD CANNOT BE RELOADED":
     PRINT " UNTIL THE SYSTEM IS REBOOTED..."
310  PRINT "ctrl-dINT"
```

13. Cambiare queste righe come segue:

```
 250  GOTO 310
 260  REM
 270  REM -- NO CARD OR CAN'T RELOAD
 280  REM
*290  IF PEEK (768)=0$ THEN 310
 300  PRINT "...LANGUAGE CARD CANNOT BE RELOADED":
      PRINT "UNTIL THE SYSTEM IS REBOOTED..."
*310  PRINT "ctrl-dRUN TURNKEY"
```

La riga 310 può anche essere messa nella forma:

```
* 310 D$="ctrl-d":PRINT D$;"RUN TURNKEY"
```

per rendere più facile la ricerca di CTRL-D

14. Memorizzare APPLESOFT (SAVE)

15. Bloccare (LOCK) APPLESOFT

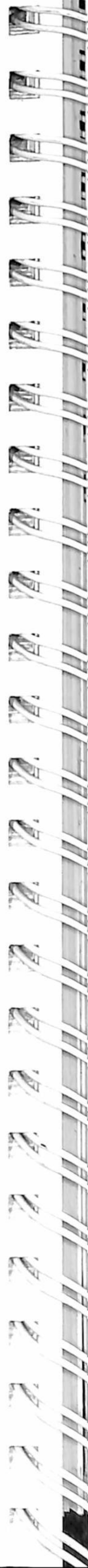
16. Ridenominare (RENAME) il programma come TURNKEY. Ecco un programma campione TURNKEY in Applesoft:

```
100  TEXT: HOME
110  VTAB 3: PRINT "DOS TOOL KIT"; TAB (30);"NOV 1979"
120  VTAB 5: PRINT "(C) COPYRIGHT 1979, APPLE COMPUTER INC."
130  FOR I=1 TO 1500: NEXT I
140  PRINT "ctrl-dRUN RIBBIT"
```

17. Assicurarsi che il mini floppy contenga i seguenti files:

```
HELLO
APPLESOFT
INTBASIC
FBASIC
TURNKEY
```

18. Provare ora il programma inserendo il mini floppy nel Drive 1 e accendendo e spegnendo l'apparecchio. Esso dovrebbe lanciare il DOS 3.3 quindi caricare Integer BASIC o Applesoft, se necessario, quindi eseguire il programma denominato TURNKEY.



APPENDICE I

Uso del mini floppy basics

Il mini floppy "BASICS" che viene fornito con il System Master DOS può essere usato per lanciare i mini floppy a 13 settori sul sistema a 16 settori. Le versioni del DOS precedenti a 3.3 sono DOS a 13 settori.

Esistono due versioni del Computer Apple : Apple II e Apple II plus. La ROM (memoria di sola lettura) della scheda principale di Apple II contiene Integer BASIC. La ROM della scheda principale di Apple II Plus contiene Applesoft. Il linguaggio nella ROM è quello che Apple utilizzerà quando viene acceso, se non dispone di unità mini floppy o schede per accessori.

Lancio dei mini floppy usando "BASICS"

Per usare Integer o Applesoft BASIC, inserire il mini floppy "BASICS" nell'unità ed accendere Apple. In circa cinque secondi, sullo schermo compare :

```
INSERT YOUR 13-SECTOR DISKETTE
AND PRESS RETURN
```

Inserire qualsiasi mini floppy DOS/BASIC a 13 settori e premere **RETURN**. Apple si comporterà ora come descritto nel manuale, salvo che alla procedura di lancio (avviamento) viene aggiunta una fase :

Quando si accende Apple, o si batte PR#6, IN#6, C6ØØG, o 6CTRL-P, il mini floppy "BASICS" deve essere nell'unità, se si vuole lanciare un DOS a 13 settori.

Quando sul video compare la richiesta, inserire il mini floppy DOS/BASIC a 13 settori che si vuole lanciare. Se si lavora con entrambi i BASIC e si batte PR#6 o IN#6, occorre inserire "BASICS" quindi reinserire il precedente mini floppy.

Una volta che il DOS è in funzione, è possibile passare da un BASIC all'altro servendosi dei comandi INT e FP.

Come sempre, è possibile sapere con quale BASIC si sta lavorando facendo riferimento al carattere di richiesta che si vede comparire sul video : > per Integer BASIC e] per Applesoft BASIC.

Un mini floppy studiato per il funzionamento personalizzato, (e cioè uno che inizia l'esecuzione del programma quando viene lanciato), può essere fatto girare come prima, dopo che è stato lanciato il mini floppy "BASICS".

È possibile rilanciare il DOS senza spegnere l'apparecchio. Basta inserire "BASICS", battere PR#6 o IN#6 e premere **RETURN** e quindi alla comparsa della richiesta, inserire il mini floppy che si vuole lanciare.

APPENDICE J

Uso del programma FID

FID (che sta per File Developer), estende le capacità del DOS a 16 settori in due modi. Innanzitutto, consente di catalogare, copiare, cancellare, bloccare e sbloccare facilmente tutti i tipi di files DOS. Secondo, consente di copiare da un mini floppy ad un altro con una sola unità mini floppy.

FID gira su un qualsiasi Apple con Applesoft, con 32 K o più di memoria ed una o più unità mini floppy Disk II. Perchè FID possa funzionare, il mini floppy "destinazione" (quello sul quale verrà eseguita la copia) deve essere inizializzato (INIT).

Come lanciare FID

Per usare FID, lanciare il mini floppy System Master DOS, quindi battere

BRUN FID

Sul video si vedrà comparire un elenco di comandi. Ciascuno di essi può essere eseguito battendo il numero tra le parentesi angolari < >.

Nomi file e caratteri per specificare parzialmente i nomi

Alcuni dei comandi chiedono un nome file, che viene battuto esattamente come un comando DOS, salvo che è possibile sostituire una parte del nome file con il carattere (=). Ad esempio, se si battesse il nome file

FR=ED

il programma selezionerebbe tutti i files i cui nomi iniziano con "FR" e terminano con "ED".
Se si battesse

=N=

verrebbero selezionati tutti i files i cui nomi contengono "N". Se si battesse

=

verrebbero selezionati tutti i files sul mini floppy ("su quale mini floppy?" ci si potrebbe chiedere. Ciò verrà spiegato più avanti).

Se è stato usato questo carattere in una specifica di nome file, verrà chiesto

DO YOU WANT PROMPTING?

Se si risponde battendo

Y

(per si), viene chiesto di verificare ciascun nome file prima di operare su di esso. Se si desidera il file, quando il sistema lo presenta sul video, battere

Y

Se non si desidera il file, battere

N

In entrambi i casi, viene richiesto conferma per il successivo nome file.

Se non si desidera il file o nulla del resto, battere

Q

ed il sistema non opererà su altri files. Se si risponde

N

alla richiesta "DO YOU WANT PROMPTING?" verranno selezionati tutti i files corrispondenti alla specifica.

I comandi

La maggior parte dei comandi FID – Catalog, Space, Delete, Lock, Unlock e Verify – operano su un'unità mini floppy : l'ultima usata, a meno che non siano state ripristinate quelle presunte. Il comando Copy può usare un'unità o due, a piacere. Il comando Quit non usa le unità mini floppy ma si limita ad uscire dal programma, lasciando l'utente in DOS.

Quando il programma compare per la prima volta, non sarà stata definita alcuna unità presunta. Se il primo comando interessa una sola unità – Catalog, Space, Delete, Lock e Unlock verrà richiesto un numero di slot e di unità, che definiranno l'unità presunta fino a che non vengono cambiati. Se il primo comando è il comando Copy, verranno richiesti due serie di numeri di Slot e di unità, una per la sorgente ed una per la destinazione. In qualsiasi momento si vuole passare da Copy ad un comando per una sola unità o viceversa, viene chiesto di specificare una o due unità. Con il comando **RESET**, è possibile cancellare in qualsiasi momento i numeri di slot e di unità presunti, il che fa si che il FID ne chieda di nuovi quando gli occorrono.

Nota : Assicurarsi di dare numeri di unità e di slot reali. Se Apple tenta di leggere da o di scrivere su un'unità inesistente, il programma si blocca.

CATALOG

Battendo

2

si ottiene un catalogo del mini floppy presunto. Se non sono stati definiti gli elementi presunti, vengono chiesti i numeri di slot e di unità. Una volta che questi sono stati definiti, tutti i commandi faranno automaticamente rifierimento a questa unità, a meno che i valori presunti non vengano cambiati.

Ripristino dello slot e dell'unità (RESET)

Il comando Reset consente di cambiare la slot e l'unità presunta. Per usarlo, battere

7

Ciò cancella i numeri di slot e di unità presunti correnti. La prossima volta che si emette un comando che richiede numeri di slot e di unità, viene chiesto di indicare tali numeri. Questo comando è comodo se si desidera catalogare (catalog) due unità mini floppy in successione.

Nota : Il comando per ripristinare lo slot e l'unità (Reset) non deve essere confuso con il tasto **RESET**. Essi saranno sempre scritti in maniera diversa. per evitare confusioni.

Blocco del file

Per bloccare (LOCK) un file, battere

5

quindi, quando compare la richiesta, battere il nome file. Se si batte un nome file non valido, la richiesta viene ripetuta fino a che non ne viene inserito uno valido.
Verrà ora chiesto di inserire il mini floppy contenente questo file e se si desidera uscire (ESC) oppure procedere. Se si procede ed il file è sul mini floppy inserito, questo verrà bloccato. Quando si esegue il catalogo del mini floppy, il nome file sarà preceduto da un asterisco, *, indicante che il file è bloccato. Se il file non è sul mini floppy, si ottiene un messaggio di errore NO FILES SELECTED; è possibile recuperare premendo qualsiasi tasto salvo **RESET**, **SHIFT**, **CTRL**, o **ESC**.

Sblocco del file

Per sbloccare un file, battere

4

e, quando compare la richiesta, battere il nome file. Se si batte un nome file non valido, la richiesta viene ripetuta fino a che non ne viene inserito uno valido.

Verrà ora chiesto di inserire il mini floppy contenente questo file e se si desidera uscire (**ESC**) o procedere. Se si procede ed il file è sul mini floppy, questo verrà sbloccato. Quando si esegue il catalogo del mini floppy, il nome file non sarà preceduto da un asterisco, *, indicando che il file non è bloccato. Se il file non è sul mini floppy, si ottiene un messaggio di erorre NO FILES SELECTED : si può recuperare premendo qualsiasi tasto salvo **RESET**, **SHIFT**, **CTRL** o **ESC**.

Verifica

Il comando Verify (verifica) è identico al comando DOS, VERIFY e gestisce tutti i tipi di files.
Per usarlo, battere

8

e rispondere alle richieste. Quando un file è stato verificato, sul video compare il suo nome e la parola

DONE

Se un file non può essere letto e pertanto non è valido, si ottiene il messaggio I/O ERROR ed il programma chiede di premere un tasto. Così facendo, si ritorna al menu principale.

Copia di un file

Il programma FID può copiare (Copy) i files su un sistema a più unità mini floppy o su un sistema a singola unità. Per copiare i file, battere

1

e si ottiene la richiesta dei numeri di slot e di unità di origine e quindi dei numeri di slot e di unità di destinazione.

Copiatura da unità mini floppy multiple

Se i numeri di slot e di unità sono diversi, viene chiesto un nome file che può contenere i caratteri=. È possibile copiare un intero mini floppy battendo= in risposta a questa richiesta. Dopo aver indicato il nome file, viene richiesto di inserire gli appropriati mini floppy e di premere **ESC**, per abortire la copiatura o qualsiasi altro tasto (salvo **RESET**, **SHIFT**, **CTRL** o **ESC**) per procedere. Se i files chiesti sono sul mini floppy di origine e se per essi c'è sufficiente spazio sul mini floppy di destinazione, si viene informati che il file ha raggiunto la sua nuova destinazione; in caso contrario si ottiene un messaggio di errore DOS.

Se il mini floppy di destinazione contiene già un file col nome che è già stato usato, si ottiene il messaggio

```
FILE
ALREADY EXISTS
TYPE IN NEW FILE NAME FOR THE COPY OR
<RETURN> TO REPLACE PRESENT FILE OR
<CTRL-C> <RETURN> TO CANCEL COPY
:
```

Se si imposta un nome file, il file verrà trasferito e memorizzato sotto quel nuovo nome. Se si preme **RETURN**, il vecchio file sarà sostituito dal nuovo, purchè il vecchio file non sia bloccato. Se lo è, si vedrà comparire:

```
FILE LOCKED.
DO YOU WISH TO REPLACE IT ANYWAY?
```

Se di batte

Y

Il file sarà sostituito;
se si batte

N

si ottiene di nuovo il messaggio:

```
FILE
ALREADY EXISTS
TYPE IN NEW FILENAME FOR THE COPY OR
<RETURN> TO REPLACE PRESENT FILE OR
<CRTL-C> <RETURN> TO CANCEL COPY
:
```

Se, quando compare questo messaggio, si preme **CTRL-C** e quindi **RETURN**, il file non verrà copiato. Se non si trovano files da copiare, il programma stampa:

NO FILES SELECTED

Se si trova un file, il programma stampa

DONE

Viene quindi chiesto :

PRESS ANY KEY TO CONTINUE

Premendo un tasto qualsiasi si ritorna al menu

Copiatura su un'unità singola

Se gli slots e le unità specificate come origine e destinazione sono le stesse, il programma dice quando inserire il disco di origine e quando inserire il disco di destinazione sull'unità. In questo modo, si possono eseguire copie tra due dischi con una sola unità.

Cancellazione dei files

Questo comando cancella (Delete) un file o una serie di files dal mini floppy presunto.
Per usarlo, battere

6

Verrà chiesto

FILENAME?

Quando si batte il nome file, il programma tenterà di cancellare il file indicato. Se tutto procede bene, si vedrà comparire sul video il messaggio

<DONE>

Se il file non può essere trovato, si ottiene il messaggio

NO FILES SELECTED

Se il file era bloccato, si ottiene il messaggio

FILE LOCKED

ed il file non viene cancellato.

Quando compare la richiesta PRESS ANY KEY TO CONTINUE, premere qualsiasi altro tasto diverso da **RETURN**, **ESC**, **CTRL**, **SHIFT** o **RESET** per ritornare al menu.

Spazio sul mini floppy

Questo comando consente di trovare quanti sono i settori usati e quanti quelli inutilizzati sul mini floppy presunto.

Per usarlo, battere

3

per far comparire tali numeri sul video.

QUIT

Il comando Quit consente di uscire dal programma in maniera semplice. Quando si batte

9

sul video si vedrà lo stesso segno di richiesta precedentemente osservato prima di eseguire

FID

Gestione degli errori

Se il programma rileva un errore irrecuperabile, presenta un messaggio di errore ed abortisce l'operazione. Se un'operazione di copiatura viene abortita, si deve fare il catalog del mini floppy di destinazione e cancellare qualsiasi file che possa essere stato creato dall'operazione. Questo file spurio, se esiste, avrà il nome del file originale, o un nuovo nome file se gliene era stato attribuito uno, ma questo sarà più corto del file originale, in quanto sarà incompleto.

Per queste condizioni vengono generati messaggi di errore : DISK FULL, DISK WRITE PROTECTED, FILE LOCKED e I/O ERROR (rispettivamente Disco pieno, Disco protetto in scrittura, File bloccato e Errore di I/O). Se compare uno di questi messaggi, il FID attende la pressione di un tasto, quindi ritorna al menu principale.

Se si verifica un errore DOS diverso da uno di quelli sopra elencati, viene stampato il suo numero di codice di errore. Ciò si verifica soltanto se è stato danneggiato il DOS o danneggiato il file o se si è tentato di copiare un BAD FILE. In uno qualsiasi di questi casi, occorre rilanciare il DOS e rieseguire (BRUN) il FID.

Se un errore provoca l'inserimento di un file spurio sul mini floppy di destinazione, occorre cancellare immediatamente quel file prima di copiare qualsiasi altro file sul mini floppy di destinazione.

APPENDICE K

Uso del programma MUFFIN

Il DOS corrente è un DOS a 16 settori. Le precedenti versioni del DOS erano a 13 settori. Per la discussione sui settori, vedere il capitolo intitolato "Uno sguardo al processo di memorizzazione" nell'Appendice C.

Un DOS a 16 settori non funziona con i mini floppy che sono stati inizializzati (INIT) con un DOS a 13 settori, in quanto sul mini floppy a 13 settori le informazioni sono disposte in maniera diversa. Se si tenta di usare il mini floppy a 13 settori con un DOS a 16 settori, si vede comparire il messaggio

    UNABLE TO READ/WRITE

Come lanciare muffin

Il programma MUFFIN converte a 16 settori i mini floppy a 13 settori ed i programmi che essi contengono, ridisponendo le informazioni e quindi scrivendole su un altro mini floppy. Il mini floppy sul quale esso scrive le informazioni (ossia il mini floppy "destinazione") deve essere stato inizializzato (INIT) con un DOS a 16 settori. Anche le informazioni dal vecchio mini floppy (il mini floppy di "origine") sono state convertite su quelle di "destinazione", la versione a 13 settori figurerà ancora sul mini floppy di origine.

Per massimizzare la quantitá di spazio sul disco utilizzabile da questo DOS, è meglio passare tutti i files a 13 settori sul mini floppy a 16 settori e quindi reinizializzare (INIT) i vecchi mini floppy con il DOS corrente.

Il programma MUFFIN può funzionare con una o con due unità mini floppy.

Questo esempio mostra come usare MUFFIN per convertire i files da un mini floppy a 13 settori in uno a 16 settori, usando un Apple con un'unità mini floppy. Si presume che l'unità disco sia collegata alla scheda unità di controllo nello slot#6.

1. Disporre il mini floppy System Master nell'unità disco e battere

    BRUN MUFFIN

2. Quando compare il messaggio

```
***************************************************************************
*                  APPLE ][ DOS 3.2 TO 3.3 CONVERTER                      *
*                                                                         *
*                                                                         *
*                        MUFFIN VERSION D                                 *
*                                                                         *
*               COPYRIGHT 1979 APPLE COMPUTER INC.                        *
***************************************************************************
CHOOSE ONE OF THE FOLLOWING OPTIONS
         <1>  CONVERT FILES
         <2>  QUIT
WHICH WOULD YOU LIKE?
```

battere

    1

per indicare che si desidera convertire i files.

3. Si vedrà ora comparire la domanda

SOURCE SLOT?

Battere 6 (il numero dello slot nel quale è inserita la scheda unità di controllo del disco). Alla successiva domanda : DRIVE? rispondere con 1.

4. Quando si vede comparire

DESTINATION SLOT?

battere 6, e quindi rispondere alla domanda DRIVE? battendo 1.

5. Verrà ora chiesto il nome del file da convertire :

FILENAME?

battere=(il segno di uguale). Il segno di uguale è un simbolo che rappresenta i nomi di tutti i files del mini floppy di origine e questa risposta significa che si desidera convertire l'intero contenuto del mini floppy d'origine.

6. Prima di poter convertire qualsiasi cosa, si vedrà comparire

DO YOU WANT PROMPTING?

Per ora battere N per no.

7. Quando si vede comparire

INSERT DISKS THEN PRESS <ESC> TO RETURN TO
MAIN MENU OR ANY OTHER KEY TO BEGIN

rimuovere il mini floppy System Master dall'unità disco e inserire un mini floppy a 13 settori contenente i files che si desidera convertire a 16 settori. Quindi premere **RETURN**.

8. Dopo la comparsa del messaggio

INSERT SOURCE DISK AND PRESS A KEY

premere semplicemente **RETURN**.

9. Il programma troverà ora il primo file sul mini floppy a 13 settori e ne stamperà il nome. Quindi si fermerà in attesa della sostituzione di quel mini floppy con un altro a 16 settori (inizializzato).
Far ciò e premere **RETURN**. Quando il file è stato convertito, si vedrà comparire

DONE

Verrà quindi chiesto di reinserire il mini floppy di origine. Ripetere questa procedura fino a che tutti i files a 13 settori non sono stati convertiti in files a 16 settori.

Nota : Per convertire file di grandi dimensioni, può essere necessario scambiare i mini floppy parecchie volte per ottenere l'intero trasferimento del file.

Quando si convertono files usando più di un'unità mini floppy, specificare i numeri di slot e di unità per i mini floppy di origine e di destinazione quando il programma li chiede (fasi 3 e 4 precedenti). Inserire i mini floppy nelle unità appropriate (fase 8) prima di iniziare la conversione.

Quando si vede comparire il messaggio

INSERT DISKS THEN PRESS <ESC> TO RETURN TO
MAIN MENU OR ANY OTHER KEY TO BEGIN

c'è la possibilità di cambiare idea a proposito della conversione del file. Se a questo punto si preme ESC, il programma si interrompe e rimanda al menu.

Se si tenta di convertire un file a 13 settori con la stesso nome del file che si trova già sul mini floppy di destinazione, si vedrà comparire :

FILE nome file
ALREADY EXISTS
TYPE IN A NEW FILE NAME FOR THE COPY
OR <RETURN>< TO REPLACE EXISTING FILE
OR <CTRL-C> <RETURN> TO CANCEL COPY :

A questo punto, si può : (a) battere un nuovo nome per il file da convertire; (b) convertire il file a 13 settori e sostituirlo al file correntemente sul mini floppy di destinazione a 16 settori sotto quel nome oppure, (c) battere **CTRL-C** e premere **RETURN** per interrompere la conversione.

Caratteri per specificare parzialmente i nomi

Come si è visto nell'esempio, il carattere=può essere usato per significare "tutti i files sul mini floppy". "=" può anche indicare qualsiasi carattere all'interno di un nome file. Ad esempio, se si risponde alla domanda

FILENAME?

battendo

FI=LE

si convertono tutti i files sul mini floppy di origine a 13 settori i cui nomi cominciano con FI e terminano con LE. Allo stesso modo=TEXT converte tutti i files i cui nomi terminano con TEXT e=*= tutti i files con i nomi contenenti un asterisco.

Comparirà la domanda

DO YOU WANT PROMPTING?

quando si usa il carattere=. Se si risponde con Y (sì), il programma si ferma dopo aver trovato ciascun file sul mini floppy a 13 settori ed attende istruzioni se convertire o meno quel file (battendo Y o N o Q per interrompere e ritornare al menu). Una risposta di N alla richiesta

DO YOU WANT PROMPTING?

indica che tutti i files nel gruppo devono essere convertiti.

Indici

Indice generale

Vedere anche l'Indice dei programmi e l'Indice dei messaggi al termine di questo Capitolo, pagina 178. All'interno della pagina di retro-copertina del manuale c'è l'Indice del sommario dei comandi e l'Indice del sommario delle procedure.

R

S

T

U

V

W

Indice dei messaggi dos

Indice dei programmi







PRINTED IN BELGIUM E2LTØ36 (Ø3Ø - T 115 - ØØ)